Electronic Colloquium on Computational Complexity, Report No. 72 (2022)

Probabilistic Kolmogorov Complexity with Applications to Average-Case Complexity

Halley Goldberg* Valentine Kabanets† Zhenjian Lu‡ Igor C. Oliveira§

May 15, 2022

Abstract

Understanding the relationship between the worst-case and average-case complexities of NP and of other subclasses of PH is a long-standing problem in complexity theory. Over the last few years, much progress has been achieved in this front through the investigation of *meta-complexity*: the complexity of problems that refer to the complexity of the input string x (e.g., given a string x, estimate its time-bounded Kolmogorov complexity). In particular, [Hir21a] employed techniques from meta-complexity to show that if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ then $\mathsf{UP} \subseteq \mathsf{DTIME}[2^{O(n/\log n)}]$. While this and related results [HN21; CHV22] offer exciting progress after a long gap, they do not survive in the setting of *randomized* computations: roughly speaking, "randomness" is the opposite of "structure", and upper bounding the amount of structure (time-bounded Kolmogorov complexity) of different objects is crucial in recent applications of meta-complexity. This limitation is significant, since randomized computations are ubiquitous in algorithm design and give rise to a more robust theory of average-case complexity [IL90].

In this work, we develop a *probabilistic* theory of meta-complexity, by incorporating randomness into the notion of complexity of a string x. This is achieved through a new probabilistic variant of time-bounded Kolmogorov complexity that we call pK^t complexity. Informally, $\mathsf{pK}^t(x)$ measures the complexity of x when shared randomness is available to all parties involved in a computation. By porting key results from meta-complexity to the probabilistic domain of pK^t complexity and its variants, we are able to establish new connections between worst-case and average-case complexity in the important setting of *probabilistic* computations:

- If $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, then $\mathsf{UP} \subseteq \mathsf{RTIME}\big[2^{O(n/\log n)}\big]$.
- If $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$, then $\mathsf{AM} \subseteq \mathsf{BPTIME}\big[2^{O(n/\log n)}\big]$.
- In the fine-grained setting [CHV22], we get $\mathsf{UTIME}[2^{O(\sqrt{n \log n})}] \subseteq \mathsf{RTIME}[2^{O(\sqrt{n \log n})}]$ and $\mathsf{AMTIME}[2^{O(\sqrt{n \log n})}] \subseteq \mathsf{BPTIME}[2^{O(\sqrt{n \log n})}]$ from stronger average-case assumptions.
- If $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$, then $\mathsf{PH} \subseteq \mathsf{BPTIME}\big[2^{O(n/\log n)}\big]$. Specifically, for any $\ell \geq 0$, if $\mathsf{Dist}\Sigma_{\ell+2}^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$ then $\Sigma_\ell^{\mathsf{P}} \subseteq \mathsf{BPTIME}\big[2^{O(n/\log n)}\big]$.
- Strengthening a result from [HN21], we show that if $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$ then polynomial size Boolean circuits can be agnostically PAC learned under any unknown $\mathsf{P/poly}$-samplable distribution in polynomial time.

In some cases, our framework allows us to significantly simplify existing proofs, or to extend results to the more challenging probabilistic setting with little to no extra effort.

*Simon Fraser University, Canada. E-mail: halley_goldberg@sfu.ca
†Simon Fraser University, Canada. E-mail: kabanets@cs.sfu.ca
‡University of Warwick, UK. E-mail: zhen.j.lu@warwick.ac.uk
§University of Warwick, UK. E-mail: igor.oliveira@warwick.ac.uk

ISSN 1433-8092

Contents

1 Introduction

1.1 Context and background

Basing the average-case hardness of NP on its worst-case hardness is one of the central questions in computational complexity theory. In the terminology of Impagliazzo's five possible complexity worlds [Imp95], this corresponds to excluding Heuristica, a world where $\mathsf{P} \neq \mathsf{NP}$ but NP problems can be efficiently solved on average with respect to every samplable distribution. Eliminating this possibility can be seen as a first step toward basing the security of cryptography on the assumption that $\mathsf{P} \neq \mathsf{NP}$.

Despite the long history of this problem (see, e.g., [Lev86]), we still have limited understanding of the relationships between the worst-case and average-case complexities of problems in NP and in other subclasses of the polynomial hierarchy. In order to understand the difficulty of making progress, a number of works have investigated limitations of known proof techniques (see, e.g., [Vio05; BT06b; Imp11]). These results suggest that fundamentally new ideas are needed to show that if NP problems are easy to solve on average, then NP is easy to solve in the worst case. For a detailed discussion on this matter, see [Hir21a, Section 1.2] and references therein.

To formally discuss the problem, we briefly review standard definitions from average-case complexity theory [BT06a]. Recall that a pair (L, D) is a *distributional problem* if $L \subseteq \{0,1\}^*$ and $D = \{D_n\}_{n\geq 1}$ is an ensemble of probability distributions, where each D_n is supported over $\{0,1\}^*$. Let $D = \{D_n\}_{n\geq 1}$ be an ensemble of this form. We say that $D \in \mathsf{PSamp}$ if there is a randomized polynomial time algorithm A such that, for every $n \geq 1$, $A(1^n)$ is distributed according to D_n. For a complexity class $\mathfrak{C}$ (e.g., $\mathfrak{C} = \mathsf{NP}$), we let $\mathsf{Dist}\mathfrak{C}$ denote the set of distributional problems (L, D) with $L \in \mathfrak{C}$ and $D \in \mathsf{PSamp}$.

We say that a distributional problem (L, D) is solvable in *polynomial time on average* if there is a (deterministic) algorithm B such that, for every n and for every x in the support of D_n, $B(x; n) = L(x)$, and there is a constant $\varepsilon > 0$ such that $\mathbf{E}_{x\sim D_n}[t_{B,n}(x)^\varepsilon] \leq O(n)$, where $t_{B,n}(x)$ denotes the running time of B on input $(x; n)$.[1] If this is the case, we write $(L, D) \in \mathsf{AvgP}$.

In a recent breakthrough, [Hir21a] established new connections between worst-case and average-complexity theory for subclasses of the polynomial hierarchy. Among other results, [Hir21a] proved that (i) if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then $\mathsf{UP} \subseteq \mathsf{DTIME}[2^{O(n/\log n)}]$;[2] and (ii) if $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgP}$, then $\mathsf{NP} \subseteq \mathsf{DTIME}[2^{O(n/\log n)}]$. While these results constitute significant progress after a long gap, they come with an important caveat: *the new connections do not hold in the setting of randomized computations*. In other words, under the assumption that NP (or $\mathsf{Dist}\Sigma_2^{\mathsf{P}}$) is easy on average for *probabilistic* algorithms (i.e., $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$), we can no longer conclude *probabilistic* worst-case upper bounds.[3]

This is an important issue for at least two reasons. On the one hand, the theory of average-case complexity is more robust when defined with respect to randomized algorithms. For instance, [IL90] proved that the average-case hardness of NP with respect to the *uniform distribution* is equivalent to its hardness with respect to the class of *samplable distributions* (see Section 2.2). On the other hand, randomized algorithms and computations are ubiquitous in both theory and practice. In

[1]It is possible to show that this is equivalent to saying that there is a constant $c > 0$ such that the probability (over D_n) that the algorithm runs for more than T steps is at most $\mathsf{poly}(n)/T^c$. We refer to [BT06a] for more information about this definition and its motivation.

[2]Recall that UP is the class of languages in NP whose positive instances admit unique witnesses.

[3]We informally discuss AvgBPP in Section 1.2.2. For a formal treatment, see Section 2.2.

particular, randomness is crucial for cryptography, which is only possible if there exist problems that are hard on average for probabilistic polynomial time algorithms (see, e.g., [Gol01]).

To explain why [Hir21a] and related works do not extend to randomized computations, we need to elaborate on their approach. These papers explore, in a crucial way, techniques from *meta-complexity*. Meta-complexity investigates the complexity of problems that refer to the complexity of the input string (e.g., given a string x, estimate its time-bounded Kolmogorov complexity). Such meta-computational problems have been at the core of other exciting recent results in complexity theory, such as a new characterization of the existence of one-way functions given in [LP20] and its subsequent developments (e.g., [RS21; LP21b]).

A central topic in meta-complexity is the study of *time-bounded Kolmogorov complexity*. Here we consider the minimum description length of a string x with respect to time-bounded machines. We informally review this notion, referring the reader to Section 2.5 for details. For a Turing machine $\mathcal{M}$, we let $|\mathcal{M}|$ denote its description length. Then, for a function $t\colon \mathbb{N} \to \mathbb{N}$ and a string $x \in \{0,1\}^*$, we let

$$\mathsf{K}^t(x) \;=\; \min_{\text{TM}\,\mathcal{M}} \{|\mathcal{M}| \mid \mathcal{M}(\varepsilon) \text{ outputs } x \text{ in } t(|x|) \text{ steps}\}\,.$$

where ε denotes the empty string. Conditional K^t complexity $\mathsf{K}^t(x \mid z)$ is defined similarly, where now the machine $\mathcal{M}$ receives z as input instead of the empty string.

Following the simplified presentation from [GK22], the approach of [Hir21a] consists of showing, under the assumption that $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgP}$, that for every $L \in \mathsf{NP}$ and every NP-verifier V for L, if $x \in L$ then some witness y_x of $x \in L$ satisfies $\mathsf{K}^t(y_x \mid x) = O(n/\log n)$, where $t(n) = 2^{O(n/\log n)}$. Consequently, in order to solve L in the worst case, it is enough to exhaustively search for a witness of complexity at most $O(n/\log n)$, which can be done deterministically in time $2^{O(n/\log n)}$. This strategy is inherently *non-black-box*, in the sense that the *code* of the average-case algorithm obtained from the initial assumption is used in a crucial way to upper bound $\mathsf{K}^t(y_x \mid x)$: it is part of the description of a machine $\mathcal{M}$ that outputs y_x.

Intuitively, if we start with the initial assumption that $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$, i.e., with a *probabilistic* average-case algorithm, the high-level approach described above simply does not work: a typical random string employed in the computation has nearly maximum Kolmogorov complexity and does not allow us to bound K^t complexity.[4] More generally, as mentioned in the abstract, "randomness" is the opposite of "structure", and upper bounding the amount of structure (time-bounded Kolmogorov complexity) of different objects is crucial in recent applications of meta-complexity.

To address this fundamental issue, we develop a suitable *probabilistic* theory of meta-complexity. In other words, we consider probabilistic notions of Kolmogorov complexity and their corresponding meta-computational problems. We are inspired in part by recent extensions of K^t complexity to the randomized setting, such as rKt complexity [Oli19; LO21] and its variant rK^t [LOS21]. Unfortunately, as explained in Section 1.3, these notions turn out to be insufficient to study relations between worst-case and average-case complexities in the setting of probabilistic computations, and a more delicate approach is necessary. To achieve this, we introduce and systematically investigate a new notion of probabilistic time-bounded Kolmogorov complexity: pK^t complexity.

[4]For the reader familiar with the arguments from [Hir21a], we mention that while it is possible to construct a pseudorandom generator under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ via [BFP05], this is not clear under the assumption that $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. See Section 1.3 for a discussion.

1.2 Results

As alluded to above, while previous works have employed various techniques to *remove* randomness from their arguments in order to analyze K^t complexity, we *incorporate* randomness in our framework. This has several advantages compared with previous works:

- We establish connections that hold in the more natural and robust setting of probabilistic computations. Previous results for deterministic computations, such as the aforementioned connections from [Hir21a], can be easily derived from our statements.
- In some contexts, we obtain significantly simpler proofs, as in the case of fine-grained connections between worst-case and average-case complexity [CHV22]. In particular, our approach does not require the design of new pseudorandom generators.
- Our probabilistic framework can improve some existing results with little to no extra effort. As a concrete example, we show how to derive agnostic learning algorithms from a weaker assumption about the average-case easiness of NP, strengthening a result from [HN21].

Next, we explain our contributions in detail. In Section 1.2.1 below, we describe the new notion of time-bounded Kolmogorov complexity that is key to our theory of probabilistic meta-complexity. Then, in Section 1.2.2, we discuss its applications to average-case complexity.

1.2.1 A new notion of probabilistic Kolmogorov complexity

Fix a function $t\colon \mathbb{N} \to \mathbb{N}$. For a string $x \in \{0,1\}^*$, the *probabilistic* t-bounded Kolmogorov complexity of x is defined as

$$\mathsf{pK}^t(x) = \min \left\{ k \;\middle|\; \mathop{\mathbf{Pr}}_{w \sim \{0,1\}^{t(|x|)}} \left[\exists\, \mathcal{M} \in \{0,1\}^k,\ \mathcal{M}(w) \text{ outputs } x \text{ within } t(|x|) \text{ steps} \right] \geq \frac{2}{3} \right\}.$$

In other words, if $\mathsf{pK}^t(x) \leq k$, then for a typical random string w, there is a (deterministic) machine $\mathcal{M}$ of length k that runs in at most $t(|x|)$ steps and prints x when given w.

Note that pK^t is conceptually different from rKt [Oli19] and rK^t [LOS21], where there is a *fixed* randomized machine $\mathcal{M}$ that outputs x with probability $\geq 2/3$ over its internal randomness. The definition of pK^t is more subtle, and its benefits are less evident. Our main conceptual discovery is that pK^t is a surprisingly useful measure of time-bounded Kolmogorov complexity.

In order to gain more intuition about this definition, consider a 2-party communication setting where a player A that knows x would like to communicate this string to a computationally bounded player B. If A and B share a typical random string w, then A can simply send to B the description of a machine $\mathcal{M}$ as above, and B will be able to run $\mathcal{M}(w)$ to recover x in at most $t(|x|)$ steps. In other words, $\mathsf{pK}^t(x)$ can be seen as $\mathsf{K}^t(x)$ in a setting where a public random string is available to all parties involved in a computation.

We elaborate now on some properties of pK^t that make it robust and particularly attractive in meta-complexity and its applications:

▪ *Short descriptions from bounded* pK^t *complexity and the complexity of a random string.* It is not immediately clear from the definition of pK^t that, in the absence of a shared random string, bounded pK^t complexity yields short descriptions (as in the case of K^t and rK^t). However, if $\mathsf{pK}^t(x) \leq k$, notice that we can *sample* x as follows: randomly generate a string w of length $t(|x|)$, randomly generate a program $\mathcal{M}$ of length k, then output whatever $\mathcal{M}(w)$ outputs after running for $t(|x|)$ steps. It is easy to see that this sampler outputs x with probability at least $\geq 2/3 \cdot 2^{-k}$. By the coding theorem for Kolmogorov complexity, it follows that x has a description of length about k. This also implies that a *random* string x of length n has $\mathsf{pK}^t(x)$ close to n, as one would expect of any reasonable and useful notion of resource-bounded Kolmogorov complexity.

- *Connection to the worst-case complexity of* NP*.* As mentioned in Section 1.1, previous results were obtained by showing, under an average-case easiness assumption, that every positive instance x admits a witness y_x such that $\mathsf{K}^t(y_x \mid x) = O(n/\log n)$, where $t(n) = 2^{O(n/\log n)}$. An important observation explored in our results is that a bound of the form $\mathsf{pK}^t(y_x \mid x) \leq k$ is also sufficient to show worst-case upper bounds. Roughly speaking, with probability $\geq 2/3$ over the choice of a random string, we can "pretend" that $\mathsf{K}^t(y_x \mid x) \leq k$, which allows us to exhaustively search for a witness in non-trivial time.

- pK^t *complexity, pseudorandom generators, and reconstruction procedures.* In a typical construction of a pseudorandom generator G based on a string x of high complexity, the correctness of G against a class of adversaries is established using a reconstruction procedure. The latter extracts from any candidate distinguisher for G an upper bound on the complexity of x. Typical reconstruction procedures are randomized, and for this reason do not yield bounds on deterministic notions of time-bounded Kolmogorov. Moreover, it is often important to fool randomized algorithms in addition to deterministic ones. As one of our key lemmas, we show that pK^t is an excellent complexity measure under these circumstances, in the sense that pK^t bounds can be obtained in a natural way from randomized distinguishers and reconstruction procedures. [5]

- *Symmetry of information for* pK^t *and average-case complexity.* We show, under the assumption $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, that pK^t satisfies the symmetry of information condition, one of the pillars of Kolmogorov complexity (see, e.g., [Lee06]). In other words, we prove under this hypothesis that for every pair of strings (x, y), $\mathsf{pK}^t(x, y) \geq \mathsf{pK}^{\mathsf{poly}(t)}(x) + \mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) - O(\log t)$. Consequently, symmetry of information is available in the probabilistic setting when investigating connections between worst-case and average-case complexity.

- *Optimal source coding theorem for* pK^t*.* As noticed by [LOZ22], pK^t admits an unconditional coding theorem with optimal parameters, the first result of this form in the time-bounded setting (see Section 3.3). This implies that if we can efficiently sample an n-bit string x with probability $\geq \delta$, then $\mathsf{pK}^{\mathsf{poly}}(x) \leq \log(1/\delta) + O(\log n)$.

- *The relationship between* pK^t*,* rK^t*, and* K^t*.* Finally, under standard derandomization assumptions, for every string $x \in \{0,1\}^n$ and constructive time bound $t(n) \geq n$, $\mathsf{K}^{\mathsf{poly}(t)}(x) = \mathsf{pK}^{\mathsf{poly}(t)}(x) = \mathsf{rK}^{\mathsf{poly}(t)}(x)$, up to an additive $O(\log t)$ term (see Section A.2 for details). In particular, results and insights about the probabilistic measure pK^t can often be translated to other previously investigated measures of time-bounded Kolmogorov complexity.

[5]Trevisan and Vadhan [TV07] observed that such reconstruction procedures are often randomized algorithms that take nonuniform advice dependent on the randomness used by the algorithm (and introduced the notation //). Our pK^t measure is a Kolmogorov complexity interpretation of the same phenomenon, with the randomness-dependent advice of [TV07] to reconstruct a string x becoming the probabilistic Kolmogorov description of x.

As a consequence of these and other desirable properties established in Section 3 (e.g., language compression for any $L \in \mathsf{AM}$ under randomized average-case easiness), pK^t is particularly well-suited for applications of meta-complexity in settings that involve probabilistic computations. In the next section, we discuss some applications of pK^t to average-case complexity.

1.2.2 Applications of pK^t to average-case complexity

A distributional problem (L, D) is in AvgBPP if it admits a randomized errorless heuristic scheme. Since the definition of a randomized heuristic scheme is somewhat technical, we refer to Section 2.2 for details, and remark that in our results the following weaker assumption suffices: there is a randomized polynomial-time algorithm B such that, for every $n \geq 1$,

(i) For every $x \in \mathsf{supp}(D_n)$, if $x \in L$, then $\mathbf{Pr}_{r_B}[B(x; n) = 1] \geq 1 - \frac{1}{n}$; and

(ii) $\mathbf{Pr}_{\substack{x \sim D_n \\ r_B}}[B(x; n) = L(x)] \geq 1 - \frac{1}{n}$,

where r_B denotes the randomness of B. In other words, our results also hold under the existence of one-sided error randomized algorithms that can make mistakes on negative instances.

First, we relate the worst-case and average-case complexities of subclasses of PH with respect to probabilistic computations.

Theorem 1 (Probabilistic Worst-Case to Average-Case Reductions)**.** *The following results hold.*

1. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{UP} \subseteq \mathsf{RTIME}\left[2^{O(n/\log n)}\right]$.
2. *If* $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{AM} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$.
3. *If* $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{PH} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$*. More specifically, for any* $\ell \geq 0$*, if* $\mathsf{Dist}\Sigma_{\ell+2}^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$*, then* $\Sigma_\ell^{\mathsf{P}} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$.

In contrast, [Hir21a] established worst-case upper bounds in $\mathsf{DTIME}[2^{O(n/\log n)}]$ assuming a corresponding inclusion in AvgP. Theorem 1 provides the first connections of this form that hold with respect to probabilistic computations.[6]

In a recent work, [CHV22] established *fine-grained* connections between worst-case and average-case complexity. For instance, they showed that if $\mathsf{NTIME}[n]$ can be (deterministically) solved in quasilinear time on average, then $\mathsf{UTIME}[2^{O(\sqrt{n \log n})}] \subseteq \mathsf{DTIME}[2^{O(\sqrt{n \log n})}]$. Next, we discuss implications of fine-grained average-case easiness assumptions in the probabilistic setting. Recall the quasilinear-time complexity classes $\mathsf{QL} = \mathsf{DTIME}[\widetilde{O}(n)]$, $\mathsf{NQL} = \mathsf{NTIME}[\widetilde{O}(n)]$, and the quasilinear-time analog $\mathsf{QLH} = \cup_{\ell \geq 0} \Sigma_\ell^{\mathsf{QL}}$ of the polynomial-time hierarchy PH; see, e.g., [NRS95] for more details.[7] Define $\mathsf{BPQL} = \mathsf{BPTIME}[\widetilde{O}(n)]$, the quasilinear-time version of BPP. Also, define QLSamp to be the class of distribution families that are quasilinear-time samplable.

Theorem 2 (Probabilistic Fine-Grained Reductions)**.** *The following results hold.*

1. *If* $\mathsf{NQL} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$*, then* $\mathsf{UTIME}\left[2^{O(\sqrt{n \log n})}\right] \subseteq \mathsf{RTIME}\left[2^{O(\sqrt{n \log n})}\right]$.

[6] As in [Hir21a], we can obtain a stronger worst-case consequence under the additional assumption that the running time of the (randomized) average-case algorithm on a given instance can be efficiently estimated (without running the algorithm). We refer to Section 4.6 for this result.

[7] As usual, we use $\widetilde{O}(T(n))$ to denote a running time of the form $O(T(n) \cdot \mathsf{poly}(\log T(n)))$.

2. *If* $\Sigma_2^{\mathsf{QL}} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$, *then* $\mathsf{AMTIME}\left[2^{O(\sqrt{n \log n})}\right] \subseteq \mathsf{BPTIME}\left[2^{O(\sqrt{n \log n})}\right]$.

In contrast with the approach of [CHV22], which requires the construction of highly efficient complexity-theoretic pseudorandom generators, our proofs are significantly simpler and do not require derandomization.

Theorem 1 and Theorem 2 are *formally stronger* than the corresponding results from [Hir21a; CHV22], which are restricted to deterministic algorithms. For instance, consider the implication from [Hir21a] that if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ then $\mathsf{UP} \subseteq \mathsf{DTIME}[2^{O(n/\log n)}]$. It immediately follows from $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, and therefore $\mathsf{UP} \subseteq \mathsf{RTIME}[2^{O(n/\log n)}]$ by Theorem 1. On the other hand, the assumption $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ yields $\mathsf{BPP} = \mathsf{P}$ [BFP05]. By padding, we get that $\mathsf{RTIME}[2^{O(n/\log n)}] = \mathsf{DTIME}[2^{O(n/\log n)}]$. (For a similar example in the fine-grained case, see the short proof of Theorem 51 in Section 4.5.)

We also remark that, similarly to previous works, we are not aware of alternate proofs of the results stated above that do not rely on (probabilistic) meta-complexity.

Next, we establish a connection between learning algorithms and randomized average-case complexity. Recall that in the PAC learning model, a learner has access to examples $(x, f(x))$ labelled according to an unknown function $f \in \mathcal{C}$, where $\mathcal{C}$ is a fixed class of Boolean functions. The examples x are drawn according to an unknown probability distribution D_n, which we assume to be supported over $\{0,1\}^n$. The goal of the learning algorithm is to produce, with high probability over its internal randomness and draw of labelled examples, a hypothesis h such that $\Pr_{x \sim D_n}[h(x) \neq f(x)] \leq \varepsilon$.

For a distribution D_n supported over $\{0,1\}^n$, we say that $D_n \in \mathsf{Samp}[T(n)]/a(n)$ if it can be sampled by an algorithm that runs in time $T(n)$ and has advice complexity $a(n)$. We consider the learnability of the class $\mathcal{C} = \mathsf{SIZE}[s]$ of Boolean circuits of size at most $s(n)$, with respect to an unknown distribution D_n from $\mathsf{Samp}[T(n)]/a(n)$. Our result holds in the more challenging setting of agnostic learning (see Section 2.6 for a review of this learning model).

Theorem 3 (Agnostic Learning from Probabilistic Average-Case Easiness of NP)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, *then for any time constructible functions* $s, T, a \colon \mathbb{N} \to \mathbb{N}$, *and* $\varepsilon \in [0,1]$, $\mathsf{SIZE}[s(n)]$ *is agnostic learnable on* $\mathsf{Samp}[T(n)]/a(n)$ *in time* $\mathsf{poly}\big(n, \varepsilon^{-1}, s(n), T(n), a(n)\big)$.

Theorem 3 strictly improves a result from [HN21], which established the same conclusion under the stronger assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$.

We finish this section with a technical remark about relativization. The fact that pK^t allows us to avoid the use of a PRG implies that our proofs relativize, i.e., the results stated above hold in the presence of any oracle. In particular, we can show that for any oracle A,

$$\mathsf{DistNP}^A \subseteq \mathsf{AvgBPP}^A \quad \Rightarrow \quad \mathsf{UP}^A \subseteq \mathsf{RTIME}^A\left[2^{O(n/\log n)}\right].$$

It is not known whether the previous (deterministic) worst-case to average-case reduction for UP, which assumes $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, holds with respect to an arbitrary oracle. More precisely, its proof depends on a PRG from [BFP05], whose proof relies on non-relativizing techniques. (In contrast, it was observed in [HN21] that one can obtain an alternate (relativized) PRG under the assumption that $\mathsf{DistP}^{\mathsf{NP}} \subseteq \mathsf{AvgP}$, so the worst-case to average-case reduction for NP, which assumes $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgP}$, relativizes.) The above implication is the best possible statement of such a theorem, as it matches the relativization barrier shown by [HN21], which says that there is an oracle $\mathcal{O}$ such that $\mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}$ but $\mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}} \not\subseteq \mathsf{BPTIME}^{\mathcal{O}}\left[2^{n/\omega(\log n)}\right]$.

1.3 Techniques

1.3.1 Hirahara's worst-case to average-case reduction

We first recall Hirahara's worst-case to average-case reduction from [Hir21a], following a presentation in [GK22]. For simplicity, we consider the case of NP only. Suppose we want an efficient worst-case algorithm for a given NP language L. The idea is to argue (under appropriate average-case easiness assumptions) that every $x \in L$ has an L-witness y_x of small conditional Kolmogorov complexity $\mathsf{K}^t(y_x \mid x)$, for a not too large time bound t, and then simply search for a good witness y by enumerating all possible short candidate Kolmogorov descriptions of y_x, decoding each in time t, and checking if it is a valid L-witness for $x \in L$. The overall time complexity of such a procedure is quasi-linear in t and exponential in $\mathsf{K}^t(y_x \mid x)$, and so we would like to minimize these two parameters.

Compression via Hadamard codes. How does one argue that a given binary string w has small K^t complexity under average-case easiness assumptions? The idea is to "encode" w into a distribution $\mathcal{D}(w)$ so that any algorithm distinguishing $\mathcal{D}(w)$ from the uniform distribution can be used to "reconstruct" w, possibly using some randomness and a few bits of advice. Several encoding methods with such properties are known in the literature on pseudorandomness. The one used by [Hir21a] is (the direct product of) the binary Hadamard error-correcting code encoding, which has an efficient list-decoding algorithm due to Goldreich and Levin [GL89]. This decoding algorithm is *randomized*, and it needs extra information (advice) about the string w in order to recover w from a distinguisher algorithm for $\mathcal{D}(w)$; moreover, the advice string a depends on the randomness r used by the decoding algorithm. To get a deterministic K^t complexity bound, Hirahara [Hir21a] fixes the random string r to be $G(\alpha)$, where G is an efficient PRG fooling polynomial-size Boolean circuits, and α is a short seed.[8] The seed α becomes part of the Kolmogorov description of w, with the other part being the advice string a corresponding to the random string $r = G(\alpha)$. Such a PRG G is known to exist under the average-case assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ [BFP05]. So one gets to upper-bound $\mathsf{K}^t(w)$, *assuming* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ *and* that one has an efficient *distinguisher* for the distribution $\mathcal{D}(w)$, where the time bound t is polynomial in the run time of the distinguisher and the length of w.

Getting a distinguisher. How does one argue the existence of an efficient distinguisher for $\mathcal{D}(w)$? Consider the case of $w = (x, y_x)$, for an n-bit string $x \in L$ (where $L \in \mathsf{NP}$) and y_x the lexicographically first L-witness for x. In the presence of the SAT oracle, one can easily compute y_x given x by a well-known search-to-decision reduction, which implies that $\mathsf{K}^{2t,\mathsf{SAT}}(x, y_x) \leq \mathsf{K}^t(x) + O(\log t)$, for any sufficiently large time bound t. On the other hand, for completely random pairs of strings (x, y), the Kolmogorov complexity of (x, y) (even with the SAT oracle) is typically larger than $\mathsf{K}^t(x) + O(\log t)$ (since y is unrelated to x). By carefully choosing the parameters of the Hadamard-based encoding of w, one gets a distinguisher for $\mathcal{D}(x, y_x)$ from uniform, where a distinguisher is a Σ_2^{P} algorithm. Since we just need a distinguisher that works well on average, the assumption $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgP}$ implies the existence of a sufficiently good deterministic polytime distinguisher for $\mathcal{D}(x, y_x)$. The latter implies that $\mathsf{K}^{\mathsf{poly}(t)}(x, y_x) \leq \mathsf{K}^t(x) + O(\log t)$.

[8]The fact that a PRG G fools the Hadamard code decoding algorithm relies on the observation that the advice string a happens to be efficiently computable from w and randomness r.

Chain rule for K^t. But we are still not done. Recall that our goal is to upperbound the conditional Kolmogorov complexity $\mathsf{K}^t(y_x \mid x)$, and so far we only have an upper bound on $\mathsf{K}^{\mathsf{poly}(t)}(x, y_x)$. Intuitively, one might hope to have a "chain rule" for K^t, saying that $\mathsf{K}^t(x,y) \approx \mathsf{K}^{t'}(x) + \mathsf{K}^{t'}(y \mid x)$, for polynomially related time bounds t and t' (such a chain rule is known for the original, time-unbounded Kolmogorov complexity, and is often called "Symmetry of Information"). It is not known if such a chain rule holds in time-bounded settings (in fact, there is some negative evidence [LW95]), but it does hold under the average-case easiness assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ [Hir21b; GK22]. Using this chain rule and the upper bound $\mathsf{K}^{\mathsf{poly}(t)}(x, y_x) \leq \mathsf{K}^t(x) + O(\log t)$, we get that $\mathsf{K}^{\mathsf{poly}(t)}(y_x \mid x) \leq \mathsf{K}^t(x) - \mathsf{K}^{\mathsf{poly}(t)}(x) + O(\log t)$.

Computational depth. Unfortunately, the two Kolmogorov complexity measures of x on the right-hand side are for different (polynomially related) time bounds, and so do not cancel out. It is still possible to get a nontrivial upper bound on such a difference (known as the computational depth of x) by making t large enough. In particular, a simple averaging argument implies that every $x \in \{0,1\}^n$ has the computational depth at most $O(n/\log n)$ for a time bound $t \leq 2^{O(n/\log n)}$. This concludes the argument bounding the conditional Kolmogorov complexity of the witness y_x.

1.3.2 Extending Hirahara's reduction to the randomized case

If Hirahara's worst-case to average-case reduction described above were *black-box*, then we could simply replace the assumed AvgP algorithm for $\mathsf{Dist}\Sigma_2^{\mathsf{P}}$ with an AvgBPP algorithm and obtain a randomized algorithm for solving every language in NP with the same running time. However, the reduction in Hirahara's argument is highly non-black-box: It crucially relies on the assumed average-case algorithm being *efficient* and *deterministic*. So we need to look inside each of the steps in the reduction and try to adapt it, using *randomized* average-case easiness assumptions.

For the Hadamard decoding step, we cannot derandomize the Goldreich-Levin algorithm as we no longer have a PRG (which is only known to exist under the *deterministic* average-case assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$). Leaving randomness in, we now get an upper bound on $\mathsf{pK}^t(w)$ from any (randomized) distinguisher for $\mathcal{D}(w)$. The fact that the advice in the Goldreich-Levin reconstruction algorithm depends on randomness forces us to use the stronger notion of randomized Kolmogorov complexity pK^t rather than rK^t. On the positive side, no average-case easiness assumptions are now needed for this step.[9]

Using a more complicated notion of pK^t creates new challenges in the next step, where we need to argue the existence of efficient distinguishers for distributions $\mathcal{D}(x, y_x)$, where y_x is the lexicographically smallest witness for $x \in L$, for some $L \in \mathsf{NP}$. Note that the pK^t definition resembles the definition of AM, where Merlin provides a Kolmogorov description of a given string, based on Arthur's randomness.[10] A naive algorithm to distinguish $\mathcal{D}(x, y_x)$ from uniform (mimicking the algorithm for the deterministic case discussed above) would be in (the promise version of) $\mathsf{AM}^{\mathsf{NP}} \subseteq \Pi_3^{\mathsf{P}}$ (see Section A.1 for a related result), which would force us to use a stronger average-case assumption like $\mathsf{Dist}\Sigma_3^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. We manage to get a good randomized distinguisher for

[9]Formally, the randomized reconstruction procedure only allows us to obtain a bound on $\mathsf{pK}^t_\delta(w)$ for $\delta = 1/\mathsf{poly}(|w|)$, where pK^t_δ is the natural generalization of pK^t with a relaxed success probability parameter δ instead of $2/3$. However, as another useful feature of pK^t complexity, we show that its success probability can be boosted with a small complexity overhead.

[10]Analogously, rK^t is similar to MA, and K^t to NP.

$\mathcal{D}(x, y_x)$ under the assumption that $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$, which is a generalization of the deterministic case from [Hir21a] discussed above. Roughly speaking, our idea is to give a random string used in the pK^t definition as an additional input to the distinguisher, reducing its complexity to Σ_2^{P}.[11]

The proof of Symmetry of Information for K^t under the assumption $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ from [GK22] is fortunately robust enough to generalize to the case of pK^t assuming $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$. We extend it further by conditioning on random strings r, and using families of strings $\{y_r\}_r$, indexed by randomness r, instead of a single string y. This allows us to get a very simple proof of a natural generalization of the worst-case to average-case reduction above to the case of AM. Namely, we show that $\mathsf{AM} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$, assuming that $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. Intuitively, for $L \in \mathsf{AM}$, witnesses y depend on both an input $x \in L$ and the randomness r used by Arthur. So it's natural to try to upperbound $\mathsf{pK}^t(y_{x,r} \mid x, r)$. However, just using (x, r) as a new input x in the original Symmetry of Information statement would result in $\mathsf{pK}^t(y_{x,r} \mid x, r)$ being upperbounded by the pK^t version of the computational depth of (x, r), which is a string of length $\mathsf{poly}(n)$, for $|x| = n$. The averaging argument for the computational depth (which works for any natural notion of Kolmogorov complexity) would give us the bound $O(n'/\log n')$ for $n' = |(x, r)| = \mathsf{poly}(n)$, which is useless. On the other hand, with the new conditional Symmetry of Information statement, we get that $\mathsf{pK}^t(y_{x,r} \mid x, r)$ is at most the *conditional* computational depth $\mathsf{pK}^t(x \mid r) - \mathsf{pK}^{\mathsf{poly}(t)}(x \mid r) + O(\log t)$. Now the input length for the averaging argument for the conditional computational depth is still $|x| = n$, yielding the upper bound $O(n/\log n)$, which allows us to prove Item 2 of Theorem 1.

While pK^t complexity is instrumental in the proof of Items 1 and 2 of Theorem 1, we show that one *can* use the simpler rK^t notion, *if* one assumes that $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$. This then allows us to get a worst-case to average-case reduction for the polytime hierarchy PH, assuming $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$, thereby proving Item 3 of Theorem 1. Another important ingredient in our PH proof is an idea of randomized *witness compression*, which allows us to perform a delicate induction on the level ℓ of PH; the randomized witness compression necessary for our inductive argument is achieved by using the rK^t complexity.

1.3.3 Fine-grained case

The use of pK^t also allows us to extend our results to the fine-grained case (Theorem 2) more easily. First of all, as observed in [CHV22], an improved worst-case running time $2^{O(\sqrt{n \log n})}$ follows if one can refine the above-mentioned computational depth expression from $\mathsf{K}^t(x) - \mathsf{K}^{\mathsf{poly}(t)}(x) + O(\log t)$ to $\mathsf{K}^t(x) - \mathsf{K}^{\widetilde{O}(t)}(x) + O(\log t)$ (note that the "blow-up" of the time bound in K^t is quasilinear in the latter as opposed to polynomial in the former). A key to such a refinement is to optimize the running time of the reconstruction procedure for the Hadamard-based encoding described above (see [CHV22, Section 2]). In particular, this running time depends on both the running time of the distinghuisher and the overhead incurred by running the PRG. It turns out that the running time of the distinghuisher, which can be obtained from a Σ_2^{QL} algorithm (in the NP case), can be optimized if one assumes that Σ_2^{QL} admits quasilinear-time average-case algorithms, which is exactly the fine-grained average-case easiness assumption. Then to minimize the overhead of the PRG, the authors of [CHV22] construct a particular efficient PRG under the fine-grained average-case easiness assumption, which incurs just a quasilinear overhead in the running time (note that this overhead would be polynomial using the PRG from [BFP05]).

[11]In the case of Theorem 1 Item 1 (i.e., for UP), we further reduce the complexity of the distinguisher so we can rely on the weaker assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$.

Thanks to the use of pK^t, we do not need any PRG in our proof! Therefore we achieve a refined probabilistic computational depth expression of $\mathsf{pK}^t(x) - \mathsf{pK}^{\widetilde{O}(t)}(x) + O(\log t)$ directly from the fine-grained probabilistic average-case easiness assumption.

1.3.4 Learning under randomized average-case assumptions

Our improvement of the learning result from [HN21] under a weaker probabilistic average-case easiness assumption, Theorem 3, follows the high-level approach from [HN21] but also crucially relies on the use of pK^t. The key idea is to design something called random-right-hand-side-refuter (RRHS-refuter). Roughly speaking, this is an efficient algorithm that distinguishes the distribution $\left(x^{(1)}, \ldots, x^{(m)}, f(x^{(1)}), \ldots f(x^{(m)})\right)$ where each $x^{(i)}$ is picked from a distribution D and f is from the concept class $\mathcal{C}$, from the distribution $\left(x^{(1)}, \ldots, x^{(m)}, b^{(1)}, \ldots b^{(m)}\right)$ where each $b^{(i)}$ is uniform. It is known that such an algorithm implies a learner for $\mathcal{C}$ under the distribution D [Vad17; KL18]. The authors of [HN21] construct a deterministic RRHS-refuter, using an algorithm that estimates the K^t complexity of a given string for a given t, which exists under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ [Hir21a]. More specifically, [HN21] shows that if a string is picked from the former case, where D is samplable by a small circuit and f is also computable by a small circuit, then it is likely to have "small" K^t complexity (for carefully chosen m and t). On the other hand, using results such as symmetry of information and optimal coding for K^t under an average-case easiness assumption [Hir21a], it can be shown that a random string from the latter case is likely to have "large" K^t complexity.

In our case, we will use a *randomized* algorithm that estimates the more complicated pK^t complexity of a given string, which we show to exist under the weaker assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$. Combining such an algorithm with the new symmetry of information for pK^t (which holds under the same probabilistic average-case easiness assumption) and an optimal coding result for pK^t from [LOZ22], we are able to construct a *randomized* RRHS-refuter, which suffices to yield the same learning result.

1.4 Directions and open problems

We have shown that several recent results proved under assumptions of the form $\mathcal{C} \subseteq \mathsf{AvgP}$ survive in the setting of randomized errorless heuristic schemes, i.e., under the weaker assumption that $\mathcal{C} \subseteq \mathsf{AvgBPP}$. Moreover, as in previous results, it is enough for us to assume the existence of *one-sided* error (randomized) algorithms that can be incorrect on negative inputs.[12] The next natural step would be to obtain results under an even weaker average-case easiness assumption such as $\mathcal{C} \subseteq \mathsf{HeurBPP}$ (i.e., under the existence of *two-sided* error randomized heuristic schemes). It would be interesting to investigate if our framework can be combined with recent results from [HS22a; HS22b] to achieve progress on this front.

We are also interested in understanding the potential of pK^t complexity. We have seen how to employ it to design agnostic learning algorithms, obtain new relations between worst-case and average-case complexity, and simplify previous proofs. Additionally, in [LOZ22] this complexity measure is used to establish an unconditional version of the main result of [AF09]. Are there more applications of pK^t to algorithms and complexity theory?

[12]Note that we also allow the average-case probabilistic algorithm to err on each fixed positive input with small probability over its internal randomness.

Finally, is it possible to extend our techniques to quantum computations, and to show that if $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBQP}$ then $\mathsf{QMA} \subseteq \mathsf{BQTIME}[2^{O(n/\log n)}]$? Exploring this and related questions is likely to lead to interesting developments in quantum time-bounded Kolmogorov complexity.

Acknowledgements. We thank Russell Impagliazzo for a clarification regarding the definition of average-case complexity classes. We are also grateful to the CCC reviewers for their comments and suggestions. This work received support from the Royal Society University Research Fellowship URF\R1\191059 and from the EPSRC New Horizons Grant EP/V048201/1. This research was also partially supported by NSERC Discovery and NSERC CGS M programs.

2 Preliminaries

2.1 Notation

We say that a function $p\colon \mathbb{N} \to \mathbb{N}$ is *non-decreasing* if $p(\ell) \geq \ell$ for every integer $\ell \geq 0$. Throughout the paper, we will rely on constructions whose associated complexity parameters (e.g., running time) can be bounded by some polynomial p. We will implicitly assume that the polynomials provided in these results are non decreasing in order to avoid including this condition in every statement. This can be done without loss of generality by the monotonicity of our bounds.

For a probability distribution D_n, we use $\mathsf{supp}(D_n)$ to denote its support. Given an element x, we let $D_n(x)$ denote the probability of x under D_n. We denote by $\mathcal{U}_n$ the uniform distribution over n-bit strings.

Let D_1 and D_2 be probability distributions supported over a set X, $A\colon X \to \{0,1\}$, and $\varepsilon \geq 0$. We say that A *ε-distinguishes* D_1 and D_2 if

$$\left| \mathop{\mathbf{Pr}}_{x \sim D_1}[A(x) = 1] - \mathop{\mathbf{Pr}}_{x \sim D_2}[A(x) = 1] \right| \;\geq\; \varepsilon.$$

If A is a randomized algorithm, each probability in the expression above also takes into account the internal randomness of A, which will be denoted by r_A.

2.2 Average-case complexity

Recall that a pair (L, D) is a *distributional problem* if $L \subseteq \{0,1\}^*$ and $D = \{D_n\}_{n \geq 1}$ is an ensemble of probability distributions, where each D_n is supported over $\{0,1\}^*$.

Let $D = \{D_n\}_{n \geq 1}$ be an ensemble of distributions. We say that $D \in \mathsf{PSamp}$ if there is a randomized polytime algorithm A such that, for every $n \geq 1$, $A(1^n)$ is distributed according to D_n. More generally, we use $D_n \in \mathsf{Samp}[T(n)]/a(n)$ to denote that D_n can be sampled by an algorithm that runs in time $T(n)$ and has advice complexity $a(n)$.

We let DistNP denote the set of distributional problems (L, D) with $L \in \mathsf{NP}$ and $D \in \mathsf{PSamp}$.

Definition 4 ($\mathsf{Avg}_\delta\mathsf{BPP}$ [BT06a])**.** Let (L, D) be a distributional problem, and $\delta\colon \mathbb{N} \to [0,1]$. We say that $(L, D) \in \mathsf{Avg}_\delta\mathsf{BPP}$ if there is randomized polytime algorithm A such that

1. for every $n > 0$, and every $x \in \mathsf{supp}(D_n)$,

$$\mathop{\mathbf{Pr}}_{r_A}[A(x; n) \notin \{L(x), \perp\}] \leq \frac{1}{4},$$

2. for every $n > 0$,

$$\Pr_{x \sim D_n}\left[\Pr_{r_A}[A(x;n) = \perp] \geq 1/4\right] \leq \delta(n),$$

where r_A denotes the internal randomness of A. Such an algorithm A is called an *randomized errorless heuristic for* (L, D) *with failure probability at most* δ.

Definition 5 (AvgBPP [BT06a][13]). We say that (L, D) is in AvgBPP if there exist a randomized algorithm A and a polynomial p such that

1. for every $n, \delta > 0$, and every $x \in \mathsf{supp}(D_n)$,

$$\Pr_{r_A}\left[A(x;n,\delta) \notin \{L(x), \perp\}\right] \leq \frac{1}{4},$$

2. for every $n, \delta > 0$,

$$\Pr_{x \sim D_n}\left[\Pr_{r_A}[A(x;n,\delta) = \perp] \geq 1/4\right] \leq \delta(n),$$

3. for every $n, \delta > 0$, and every $x \in \mathsf{supp}(D_n)$, $A(x;n,\delta)$ runs in time at most $p(n/\delta)$.

Such an algorithm A is called a *randomized errorless heuristic scheme for* (L, D).

Lemma 6. *Let* (L, D) *be a distributional problem in* $\mathsf{Avg}_{n^{-1}}\mathsf{BPP}$. *There exists a randomized polynomial-time algorithm* B *such that*

1. *for every* $n > 0$, *and every* $x \in \mathsf{supp}(D_n)$, $B(x;n) \in \{0, 1\}$,
2. *for every* $n > 0$, *and every* $x \in \mathsf{supp}(D_n)$, *if* $x \in L$, *then* $\Pr_{r_B}[B(x;n) = 1] \geq 1 - \frac{1}{n}$, *and*
3. *for every* $n > 0$, $\Pr_{\substack{x \sim D_n \\ r_B}}[B(x;n) = L(x)] \geq 1 - \frac{3}{n}$.

Proof. Let A be an $\mathsf{Avg}_{n^{-1}}\mathsf{BPP}$ algorithm for (L, D) as in Definition 4. Let A' be the algorithm obtained, using standard amplification techniques,[14] such that for all $x \in \mathsf{supp}(D_n)$,

$$\Pr_{r_{A'}}[A'(x;n) \notin \{L(x), \perp\}] \leq \frac{1}{n} \tag{1}$$

and

$$\Pr_{x \sim D_n}\left[\Pr_{r_{A'}}[A'(x;n) = \perp] \geq 1/n\right] \leq \frac{1}{n}. \tag{2}$$

Let B be the randomized algorithm which, on input $x \in \mathsf{supp}(D_n)$, simulates $A'(x;n)$ and outputs 1 if $A'(x;n) \in \{1, \perp\}$ or 0 if $A'(x;n) = 0$. By Equation (1), if $x \in L$, then

$$\Pr_{r_B}[B(x;n) = 0] = \Pr_{r_{A'}}[A'(x;n) \notin \{L(x), \perp\}] \leq \frac{1}{n}.$$

[13] Some authors define AvgBPP as $\cap_{c>0}\mathsf{Avg}_{n^{-c}}\mathsf{BPP}$. All our results also hold with respect to that definition.

[14] In more detail, A' can be obtained from A by running it $O(n)$ times and selecting the most common output.

Let $bad := \{x \in \mathsf{supp}(D_n) \mid \mathbf{Pr}_{r_{A'}}[A'(x;n) = \bot] \geq 1/n\}$. By Equation (2), at most a $(1/n)$-measure of $x \in \mathsf{supp}(D_n)$ are in bad. Thus,

$$\begin{aligned}
\mathop{\mathbf{Pr}}_{\substack{x\sim D_n\\ r_B}}[B(x;n) \neq L(x)] &\leq \mathop{\mathbf{Pr}}_{x,r_{A'}}[A'(x;n) = \bot] + \mathop{\mathbf{Pr}}_{x,r_{A'}}[A'(x;n) \notin \{L(x), \bot\}] \\
&\leq \mathop{\mathbf{Pr}}_{x}[x \in bad] + \mathop{\mathbf{Pr}}_{x,r_{A'}}[A'(x;n) = \bot \mid x \notin bad] + \mathop{\mathbf{Pr}}_{x,r_{A'}}[A'(x;n) \notin \{L(x), \bot\}] \\
&\leq \frac{1}{n} + \frac{1}{n} + \frac{1}{n} = \frac{3}{n}.
\end{aligned}$$

This completes the proof. □

The following result of Impagliazzo and Levin [IL90] shows that, for every $\ell \geq 1$, the easiness of $\mathsf{Dist}\Sigma_\ell^{\mathsf{P}}$ is equivalent to that of $(\Sigma_\ell^{\mathsf{P}}, \mathcal{U})$. This implies that the average-case easiness under uniform distribution is sufficient for all our main results.

Theorem 7 ([IL90]; see also [BT06a, Theorem 31][15])**.** *For every $\ell \geq 1$,*

$$\left(\Sigma_\ell^{\mathsf{P}}, \mathsf{PSamp}\right) \subseteq \mathsf{AvgBPP} \iff \left(\Sigma_\ell^{\mathsf{P}}, \mathcal{U}\right) \subseteq \mathsf{AvgBPP}.$$

2.3 Pseudodeterministic PRGs

Definition 8 (Pseudodeterministic PRG [OS17; LOS21])**.** Fix an error function $\varepsilon\colon \mathbb{N} \to \mathbb{R}$. A function family $G_n\colon \{0,1\}^\ell \to \{0,1\}^n$ is an ε*-error Pseudorandom Generator (PRG)* if, for all sufficiently large $n \in \mathbb{N}$, the distribution $G(\mathcal{U}_{\ell(n)})$ $\varepsilon(n)$-fools circuits of size n, using the seed length $\ell = \ell(n)$, i.e., for all circuits C of size n on n inputs,

$$\left| \mathop{\mathbf{Pr}}_{r\sim\{0,1\}^n}[C(r) = 1] - \mathop{\mathbf{Pr}}_{\alpha\sim\{0,1\}^{\ell(n)}}[C(G(\alpha)) = 1] \right| \leq \varepsilon.$$

Such a PRG G is called a *pseudodeterministic PRG* if there is a randomized algorithm M_G such that, for every n and $\alpha \in \{0,1\}^{\ell(n)}$, $M_G(1^n, \alpha)$ outputs the string $G_n(\alpha) \in \{0,1\}^n$ with probability at least $1 - 2^{-n}$ over its internal randomness, and M_G runs in time $2^{O(\ell(n))}$.

Lemma 9 ([IW97; OS17])**.** *If* BPE *contains a language L of circuit complexity $2^{\Omega(n)}$ for almost all input lengths n, then there is an ε-error pseudodeterministic PRG of seed length $O(\log n/\varepsilon)$.*

Lemma 10 ([MVW99, Lemma 2])**.** $\mathsf{E}^{\Sigma_2^{\mathsf{P}}}$ *contains a language of maximum circuit complexity (at least $2^n/n$) for almost all input lengths n.*

Using padding as in [Ben+92], we get the following.

Lemma 11. *If* $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{E}^{\Sigma_2^{\mathsf{P}}} = \mathsf{BPE}$*.*

[15] The published paper by Impagliazzo and Levin [IL90] contains a proof applicable only to the case of heuristics with errors (such as $\mathsf{HeurBPP}$). The case of errorless heuristics (such as AvgBPP) requires a different argument, which is given in Section 5.2 of [BT06a], based on Impagliazzo's unpublished notes.

Proof. The inclusion $\mathsf{BPE} \subseteq \mathsf{E}^{\Sigma_2^\mathsf{P}}$ follows from the inclusion $\mathsf{BPP} \subseteq \Sigma_2^\mathsf{P}$ [Lau83] by padding. For the other direction, let $L \in \mathsf{E}^{\Sigma_2^\mathsf{P}}$ be arbitrary. Define its padded version $L^{pad} = \{(x, 1^{2^{2|x|}}) \mid x \in L\}$. Note that $L^{pad} \in \mathsf{P}^{\Sigma_2^\mathsf{P}}$. Define a family of distributions $D_n = (\mathcal{U}_n, 1^{2^{2n}})$, for the uniform distribution over n-bit strings $\mathcal{U}_n$. By assumption, $(L^{pad}, D_n) \in \mathsf{AvgBPP}$, and so, by Lemma 6, Item 3, there is a randomized polytime algorithm B such that $B(x, 1^{2^{2|x|}}) \neq L(x)$ with probability less than $3 \cdot 2^{-2n} < 1/(3 \cdot 2^n)$, where the probability is over a uniformly random $x \in \{0,1\}^n$ and the internal randomness of B. It follows that, for *every* $x \in \{0,1\}^n$, $B(x, 1^{2^{2|x|}}) \neq L(x)$ with probability less than $1/3$ over its internal randomness. Hence, the randomized algorithm $B'(x) := B(x, 1^{2^{2|x|}})$ decides if $x \in L$ with probability at least $2/3$, for all sufficiently large inputs $x \in \{0,1\}^n$, yielding $L \in \mathsf{BPE}$. $\square$

Corollary 12. *If $\mathsf{DistP}^{\Sigma_2^\mathsf{P}} \subseteq \mathsf{AvgBPP}$, then there is an ε-error pseudodeterministic PRG of seed length $O(\log n/\varepsilon)$.*

Proof. Immediate by combining Lemmas 9 to 11. $\square$

2.4 Direct Product Generator

For $x, z \in \{0,1\}^n$, we let $x \cdot z := \sum_{i=1}^n x_i z_i \pmod 2$ denote their inner product modulo 2.

Definition 13 (Direct Product Generator (DPG) [Hir21a, Definiton 3.10])**.** For $k, n \in \mathbb{N}$, we define the *k-wise direct product generator* to be the function

$$\mathsf{DP}_k \colon \{0,1\}^n \times \{0,1\}^{nk} \to \{0,1\}^{nk+k}$$

such that

$$\mathsf{DP}_k(x; z^1, \ldots, z^n) := (z^1, \ldots, z^k, x \cdot z^1, \ldots, x \cdot z^k).$$

Lemma 14 (DPG Reconstruction [Hir21a, Lemma 3.14])**.** *For any $n, k \in \mathbb{N}$ with $k \leq 2n$, and $\varepsilon > 0$, there exists a pair of algorithms A and $\mathsf{Recon}^{(-)}$ such that*

- *Recon^D takes oracle access to an oracle $D \colon \{0,1\}^{nk+k} \to \{0,1\}$.*
- *$A \colon \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^k$ is called the advice function and is computable in time $\mathsf{poly}(n/\varepsilon)$.*
- *$\mathsf{Recon}^{(-)} \colon \{0,1\}^k \times \{0,1\}^r \to \{0,1\}^n$ is called a reconstruction procedure and is computable in time $\mathsf{poly}(n/\varepsilon)$.*[16]
- *The randomness complexity r is at most $\mathsf{poly}(n/\varepsilon)$.*
- *For any string $x \in \{0,1\}^n$ and any function D that ε-distinguishes $\mathsf{DP}_k(x; \mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$, it holds that*

$$\mathop{\mathbf{Pr}}_{w \sim \{0,1\}^r}\left[\mathsf{Recon}^D(A(x,w), w) = x\right] \geq 1/\mathsf{poly}(n/\varepsilon).$$

[16]We note that if the oracle is a uniform algorithm that runs in time $t(m)$ on inputs of length m, the reconstruction procedure can be computed in time $t(nk+k) \cdot \mathsf{poly}(n/\varepsilon)$.

2.5 Kolmogorov complexity measures

For a string $w \in \{0,1\}^*$, we use $|w| \in \mathbb{N}$ to denote its length. We let ϵ represent the empty string. Let U be a Turing machine. For a function $t \colon \mathbb{N} \to \mathbb{N}$ and a string $x \in \{0,1\}^*$, we let

$$\mathsf{K}_U^t(x) \overset{\text{def}}{=} \min_{p \in \{0,1\}^*} \Big\{ |p| \mid U(p,\epsilon) \text{ outputs } x \text{ in at most } t(|x|) \text{ steps} \Big\}$$

be the *t-time-bounded Kolmogorov complexity* of x. The machine U is said to be *time-optimal* if for every machine M there exists a constant c_M such that for all $x \in \{0,1\}^n$ and $t \colon \mathbb{N} \to \mathbb{N}$ satisfying $t(n) \geq n$,

$$\mathsf{K}_U^{c_M \cdot t \log t}(x) \leq \mathsf{K}_M^t(x) + c_M,$$

where for simplicity we write $t = t(n)$. It is well known that there exist time-optimal machines (see, e.g., [LV19a, Chapter 7]). We fix such a machine, and drop the index U when referring to time-bounded Kolmogorov complexity measures.

We remind the reader that the (time-unbounded) *Kolmogorov complexity* of a string x, denoted $\mathsf{K}(x)$, is defined in the same way but does not impose a fixed upper bound $t(|x|)$ on the running time of $U(p, \varepsilon)$.

Given strings $x, y \in \{0,1\}^*$, we can also consider the *conditional t-time-bounded Kolmogorov complexity of x given y*, defined as

$$\mathsf{K}^t(x \mid y) \overset{\text{def}}{=} \min_{p \in \{0,1\}^*} \Big\{ |p| \mid U(p,y) \text{ outputs } x \text{ in at most } t(|x|) \text{ steps} \Big\}.$$

From now on, we will not distinguish between a Turing machine M and its encoding p_M according to U. While the running time t of M on an input y and the running time of the universal machine U on (p_M, y) might differ by a multiplicative factor of $O(\log t)$, this will be inessential in all our results.[17] For simplicity, we will assume the encoded machines to U are paddable so that M and $M \circ 0^a$ denote the same machine for every natural number a.

We denote by $\mathsf{rK}_\delta^t(x)$ the minimum length of a randomized machine $\mathcal{M}$ that outputs x with probability at least δ when running for at most t steps (see [Oli19; LOS21]). We simply write $\mathsf{rK}^t(x)$ when $\delta = 2/3$.

We formally introduce $\mathsf{pK}_\delta^t(x)$ and discuss its properties in Section 3.

2.6 Agnostic learning model

This section reviews standard notions from computational learning theory (see, e.g., [SB14]). Consider the problem of learning an unknown *concept* $f \colon X \to \{0,1\}$ over a finite *domain* X. We will always let $X = \{0,1\}^n$, for some $n \geq 1$. A *concept class* $\mathcal{C}$ is a collection of concepts of this form. For a fixed n, we let $\mathcal{C}_n$ denote $\mathcal{C} \cap \{f \colon \{0,1\}^n \to \{0,1\}\}$. For a size function $s \colon \mathbb{N} \to \mathbb{N}$, we let

$$\mathsf{SIZE}[s] = \{f \colon \{0,1\}^n \to \{0,1\} \mid n \in \mathbb{N} \text{ and } f \text{ admits a size-}s(n) \text{ circuit}\}$$

denote the concept class of Boolean circuits of size (number of gates) at most $s(n)$ over n input variables. For definiteness, we consider circuits over AND and OR gates of fan-in 2 and NOT gates.

[17] It is also possible to consider prefix-free notions of Kolmogorov complexity. Since our results hold up to additive $O(\log |x|)$ terms, we will not make an explicit distinction.

A *randomized Boolean function* f maps an input x to a distribution D supported over $\{0,1\}$. Given a distribution D_n supported over $\{0,1\}^n$, a function $h\colon \{0,1\}^n \to \{0,1\}$, and a (possibly randomized) Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, we let

$$\mathsf{err}_{D_n}(h,f) = \Pr_{f,\, x \sim D_n}[h(x) \neq f(x)].$$

We will consider the (agnostic) learnability of a concept class $\mathcal{C}$ with respect to a *class* $\mathcal{D}$ of ensembles $D = \{D_n\}_{n \geq 1}$ of distributions D_n. We use $\mathcal{D}_n$ to denote $\{D_n \mid D \in \mathcal{D}\}$. For simplicity, we refer to $\mathcal{D}$ as a class of distributions.

Given a randomized function $f\colon \{0,1\}^n \to \{0,1\}$ and a distribution D_n, the learning algorithm has access to an *example oracle* $\mathsf{EX}(f, D_n)$ that behaves as follows: each query to $\mathsf{EX}(f, D_n)$ returns an independent and identically distributed pair (x, b), where x is sampled according to D_n and $b = f(x)$. The *sample complexity* of the learning algorithm is the number of queries made to $\mathsf{EX}(f, D_n)$.

Before formalising the agnostic learning model, we provide an informal overview. The learning algorithm *knows* the class $\mathcal{D}$ of distributions and the concept class $\mathcal{C}$. It is given access to $\mathsf{EX}(f, D_n)$ for some *unknown* $D \in \mathcal{D}_n$ and an *arbitrary* (possibly randomized) function $f\colon \{0,1\}^n \to \{0,1\}$. The goal of the learning algorithm is to produce, with high probability over its internal randomness and queries to $\mathsf{EX}(f, D)$, a hypothesis h such that $\mathsf{err}_D(h, f) \leq \mathsf{opt}_{\mathcal{C}_n, D, f} + \varepsilon$, where

$$\mathsf{opt}_{\mathcal{C}_n, D, f} \;=\; \min_{c \in \mathcal{C}_n} \; \mathsf{err}_D(c, f).$$

In other words, the learning algorithm is required to output an efficient representation of a hypothesis $h\colon \{0,1\}^n \to \{0,1\}$ that is almost as accurate as the best concept in $\mathcal{C}_n$ with respect to the pair (D, f). (Note that h is not required to be a function in $\mathcal{C}$.)

Definition 15 (Agnostic PAC learning [KSS94])**.** Let $\mathcal{C}$ be a concept class, and let $\mathcal{D}$ be a class of distributions. We say that a randomized algorithm A agnostically learns $\mathcal{C}$ on $\mathcal{D}$ if the following holds. For every $n \geq 1$, distribution $D \in \mathcal{D}_n$, and randomized function $f\colon \{0,1\}^n \to \{0,1\}$, when A is given oracle access to $\mathsf{EX}(f, D)$ and receives as input n, $\varepsilon > 0$, and $\delta > 0$,

$$\Pr_{A,\, \mathsf{EX}(f,D)}\left[A^{\mathsf{EX}(f,D)} \text{ outputs a hypothesis } h \text{ such that } \mathsf{err}_D(h,f) \leq \mathsf{opt}_{\mathcal{C}_n, D, f} + \varepsilon \right] \;\geq\; 1 - \delta.$$

We remark that an equivalent way of formulating the agnostic learning model is by considering distributions $\mathcal{D}$ supported over $\{0,1\}^{n+1}$. In this case, there is no need to refer to randomized functions, and one measures the error of a hypothesis h via $\mathsf{err}_{\mathcal{D}}(h) = \Pr_{(x,b) \sim \mathcal{D}}[h(x) \neq b]$, where $x \in \{0,1\}^n$ and $b \in \{0,1\}$.

2.7 Input encodings

In some situations, it will be useful to refer to collections of probability distributions that are indexed by a constant number of parameters. It is not hard to reduce this to the case of a single parameter 1^n. Moreover, it is possible to assume without loss of generality that distribution D_n is supported over $\{0,1\}^n$ instead of $\{0,1\}^{\leq \mathsf{poly}(n)}$. This can be done using standard techniques, and will be implicitly assumed in our arguments. In any case, for completeness, we include more details about input encodings in this section.

For $j \in \mathbb{N}$, we use $\mathsf{bin}(j) \in \{0,1\}^{\lfloor \log j \rfloor + 1}$ to denote its binary representation. We will often consider languages that view their inputs as a tuple of parameters. We describe how to encode such an input into a single binary string. For a tuple $(a_1, a_2, \dots, a_k)$ where $k \in \mathbb{N}$, we encode it as

$$\ell_1 \circ 01 \circ \ell_2 \circ 01 \circ \ell_{k-1} \circ 01 \circ a_1 \circ a_2 \circ \cdots \circ a_k,$$

where for each $i \in [k]$, ℓ_i is obtained from $\mathsf{bin}(|a_i|)$ with every bit doubled. Note that in this case, $(a_1, a_2, \dots, a_k)$ has an encoding length of

$$|a_k| + \sum_{i=1}^{k-1} \left(2 \cdot (\lfloor \log |a_i| \rfloor + 2) + |a_i|\right). \tag{3}$$

We will also need to consider distributions that generate input instances that are tuples, while the only known information to the distributions is the length of the instances. To do this, we will encode the format as the length of the instances. More specifically, we use the following way to encode tuples [BT06a].

Proposition 16 ([BT06a]; see also [CHV22, Proposition 3.1])**.** *There is a pair of polytime encoding-decoding algorithms* $\mathsf{Enc}\colon \mathbb{N}^* \to \mathbb{N}$ *and* $\mathsf{Dec}\colon \mathbb{N} \to \mathbb{N}^* \cup \{\bot\}$ *such that*

1. *For $k \in \mathbb{N}$ and k integers $n_1, n_2, \dots, n_k$, we write $\langle n_1, n_2, \dots, n_k \rangle := \mathsf{Enc}(n_1, n_2, \dots, n_k)$, and we have*
   - *(a)* $\sum_{i=1}^{k} (2 \cdot (\lfloor \log n_i \rfloor + 2) + n_i) \le \langle n_1, n_2, \dots, n_k \rangle \le O_k\big(\Pi_{i=1}^{k} (n_i \cdot \log^2 n_i)\big)$.
   - *(b)* $\mathsf{Dec}(\langle n_1, n_2, \dots, n_k \rangle) = n_1, \dots, n_k$. *Here,* Dec *takes the binary representation of an integer and outputs k integers which are also represented in binary.*
2. $\mathsf{Dec}(u) = \bot$ *if $u \neq \langle \vec{n} \rangle$ for any $\vec{n} \in \mathbb{N}^*$.*

Proof. Given k integers $n_1, n_2, \dots, n_k$, the encoding algorithm Enc first obtains the binary string

$$z := \alpha_1 \circ 01 \circ \alpha_2 \circ 01 \circ \alpha_k \circ 01 \circ \mathsf{bin}(n_1) \circ \mathsf{bin}(n_2) \circ \cdots \circ \mathsf{bin}(n_k),$$

where for each $i \in [k]$, α_i is obtained from $\mathsf{bin}(|\mathsf{bin}(n_i)|)$ with every bit doubled. Then Enc outputs the integer n whose binary representation is z. It is easy to see that given the binary representation of n, which is simply the string z, one can recover the $n_1, n_2, \dots, n_k$ in time polynomial in $|z|$. Also, item (a) can be verified by a simple calculation. □

We give an example of how we will typically use Proposition 16. Suppose we have some language L that takes $k = 3$ parameters, then we may define a family of distributions $\{D_n\}_n$, where each D_n does the following.

1. On input 1^n, if $\mathsf{Dec}(n) \neq n_1, n_2, n_3$ for any $n_1, n_2, n_3 \in \mathbb{N}$, then output 0^n.
2. Otherwise, sample $x \sim \{0,1\}^{n_1}$, $y \sim \{0,1\}^{n_2}$, and let $z := 1^{n_3}$.
3. Output $\left(x, y, z, 0^{n - \sum_{i=1}^{3} (2 \cdot (\lfloor \log n_i \rfloor + 2) + n_i)}\right)$.

Note that D_n runs in time $\mathsf{poly}(n)$, and the output length (using Equation (3)) is

$$\left(n - \sum_{i=1}^{3} (2 \cdot (\lfloor \log n_i \rfloor + 2) + n_i)\right) + \sum_{i=1}^{3} (2 \cdot (\lfloor \log n_i \rfloor + 2) + n_i) = n.$$

Also, note that the above distribution D_n outputs instances with $k+1$ parameters, where the last parameter is just some padding so that the output is of length n. To cope with this, instead of the language L, we will then work with a padded language L' which takes $k+1$ parameters and L' simulates L by ignoring the last parameter. In particular, for some $\ell_1, \ell_2, \ell_3 \in \mathbb{N}$ of interest, we can use $D_{\langle \ell_1, \ell_2, \ell_3 \rangle}$ to generate instances of the form (x, y, z, g) where $x \in \{0,1\}^{\ell_1}, y \in \{0,1\}^{\ell_1}, z \in \{0,1\}^{\ell_3}$, and the length of such instances is $O\big(\Pi_{i=1}^{3} \ell_i \cdot \mathsf{polylog}(\ell_i)\big) = \widetilde{O}(\ell_1 \cdot \ell_2 \cdot \ell_3)$.

3 Probabilistic Time-Bounded Kolmogorov Complexity

In this section, we formally define probabilistic Kolmogorov complexity and prove some useful properties of this notion, which will be used later in proving our main results.

Definition 17 (pK^t)**.** For strings $x, y \in \{0,1\}^*$, a time bound $t \in \mathbb{N}$, an oracle A, and $\delta \in [0,1]$, the δ-probabilistic A-oracle t-bounded Kolmogorov complexity of x given y is defined as

$$\mathsf{pK}_{\delta}^{t,A}(x \mid y) := \min \left\{ k \;\middle|\; \Pr_{w \sim \{0,1\}^t} \Big[\exists \mathcal{M} \in \{0,1\}^k \text{ s.t. } \mathcal{M}^A(w,y) \text{ outputs } x \text{ within } t \text{ steps} \Big] \geq \delta \right\},$$

where $\mathcal{M}$ is a RAM-machine.[18] We omit the subscript δ if $\delta = 2/3$, omit the superscript A if $A = \varnothing$, omit "$\mid y$" if y is the empty string, and omit the superscript t if $t = \infty$.

For simplicity, in the rest of this section we only consider pK^t without oracles. It is easy to see that all the results also hold with any oracle.

3.1 Basic properties of pK^t complexity

Lemma 18. *There is a universal constant $c > 0$ such that the following holds. For every time bound $t \in \mathbb{N}$ and $x \in \{0,1\}^n$,*

$$\mathsf{K}(x \mid t) \leq \mathsf{pK}^t(x) + c \log n.$$

Proof. Let $\ell := \mathsf{pK}^t(x)$. Since x is of length n, we can represent ℓ using $O(\log n)$ bits. Given t, ℓ and n, it is possible to *sample* x with probability $\mu \geq 2/3 \cdot 2^{-\ell}$ by randomly guessing $w \sim \{0,1\}^t$ and $\mathcal{M} \sim \{0,1\}^\ell$ then simulating $\mathcal{M}(w)$ for t steps. Consequently, by the coding theorem for (time-unbounded) Kolmogorov complexity [Lev74], we get that $\mathsf{K}(x \mid t) \leq \log(1/\mu) + O(\log n) \leq \mathsf{pK}^t(x) + O(\log n)$. $\square$

[18]We use the RAM model in this definition for convenience: it simplifies the time bound estimates for a machine $\mathcal{M}$ that takes as input both randomness w and an "auxiliary" string y, and allows both w and y to be as long as the running time of the RAM-machine $\mathcal{M}$; cf. [LP21a] for a similar definition in the case of deterministic time-bounded conditional Kolmogorov complexity. Alternatively, we could use a TM model, with inputs w and y presented on two different tapes. However, we prefer the RAM model as it's also useful in the context of fine-grained worst-case to average-case reductions [CHV22].

Proposition 19. *For any string $x \in \{0,1\}^*$, time bound $t \in \mathbb{N}$, and $\delta \in [0,1]$, we have*

$$\mathsf{pK}^t_\delta(x) \le \mathsf{rK}^t_\delta(x) \le \mathsf{K}^t(x).$$

Proof. Immediate from the definition of the involved Kolmogorov complexity measures. ☐

Lemma 20 (Probabilistic Incompressibility)**.** *For any string $y \in \{0,1\}^*$, time bound $t \in \mathbb{N}$ (including $t = \infty$), $\delta \in (0,1]$, and positive integer α, we have*

$$\Pr_{x \sim \{0,1\}^n}\left[\mathsf{pK}^t_\delta(x \mid y) \le n - \alpha\right] \le \frac{1}{2^{\alpha-1} \cdot \delta}.$$

Proof. For the sake of contradiction, suppose the statement of the lemma does not hold. Then by the definition of pK^t_δ, we have

$$\mathop{\mathbf{E}}_{\substack{x \sim \{0,1\}^n \\ w \sim \{0,1\}^t}}\left[\exists\, \mathcal{M}_{(x,w)} \in \{0,1\}^{\le n-\alpha} \text{ such that } \mathcal{M}_{(x,w)}(w,y) \text{ outputs } x \text{ within } t \text{ steps}\right] > \frac{1}{2^{\alpha-1}}.$$

By averaging, there is a way to fix $w \in \{0,1\}^t$ that at least preserves this expectation, yielding

$$\Pr_{x \sim \{0,1\}^n}\left[\exists\, \mathcal{M}_{(x,w)} \in \{0,1\}^{\le n-\alpha} \text{ such that } \mathcal{M}_{(x,w)}(w,y) \text{ outputs } x \text{ within } t \text{ steps}\right] > \frac{1}{2^{\alpha-1}}.$$

However, by counting, the above probability is at most $2^{n-\alpha+1}/2^n = 1/2^{\alpha-1}$. A contradiction. ☐

Lemma 21 (Success Amplification)**.** *For any string $x \in \{0,1\}^n$, time bound $t \in \mathbb{N}$, and $0 \le \alpha < \beta \le 1$, we have*

$$\mathsf{pK}^{O(qt/\alpha)}_\beta(x) \le \mathsf{pK}^t_\alpha + O(\log(q/\alpha)),$$

where $q = \ln(1/(1-\beta))$.

Proof. Suppose $\mathsf{pK}^t_\alpha(x) = k$. Then by definition, we have

$$\Pr_{w \sim \{0,1\}^t}\left[\exists\, \mathcal{M}_w \in \{0,1\}^k \text{ such that } \mathcal{M}_w(w) \text{ outputs } x \text{ within } t \text{ steps}\right] \ge \alpha.$$

Call such a w *good*. After sampling ℓ independent $w^{(1)}, w^{(2)}, \ldots, w^{(\ell)} \in \{0,1\}^t$, the probability that no $w^{(i)}$ is good is at most $(1-\alpha)^\ell \le e^{-\alpha\ell}$, which can be made at most $1-\beta$ by choosing $\ell = q/\alpha$. It follows that, with probability at least β over a random $w = \left(w^{(1)}, w^{(2)}, \ldots, w^{(q/\alpha)}\right) \in \{0,1\}^{qt/\alpha}$, there exists an index $1 \le i \le q/\alpha$ (described with at most $\lceil \log(q/\alpha) \rceil$ bits) and a program $\mathcal{M}_{w^{(i)}} \in \{0,1\}^k$ such that $\mathcal{M}_{w^{(i)}}\left(w^{(i)}\right)$ outputs x within $O(qt/\alpha)$ steps. Hence, $\mathsf{pK}^{O(qt/\alpha)}_\beta(x) \le k + O(\log(q/\alpha))$. ☐

Note that the parameter α affects the time complexity in the resulting $\mathsf{pK}^{O(qt/\alpha)}_\beta$ bound. Lemma 21 is sufficient in applications where $\alpha \ge 1/\mathsf{poly}(n)$.

3.2 Bounding pK^t and rK^t via DPG reconstruction

The following is a key lemma for pK^t that we will use in proving our main results related to randomized average-case complexity.

Lemma 22 (pK^t Reconstruction Lemma)**.** *For $\varepsilon > 0$, $x \in \{0,1\}^n$, $s \in \mathbb{N}$, and $k \in \mathbb{N}$ satisfying $k \leq 2n$, let D be a randomized algorithm that takes an advice string β and runs in time t_D such that D ε-distinguishes $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Then there is a polynomial p_{DP} such that*

$$\mathsf{pK}^{\widetilde{O}(t_D)\cdot p_{\mathsf{DP}}(n/\varepsilon)}(x \mid \beta) \leq k + \log p_{\mathsf{DP}}(nt_D/\varepsilon).$$

Proof. We will view the distinguisher D as a deterministic algorithm that takes a string of t_D bits, denoted by r_D, as its internal randomness. By our assumption (and dropping the absolute value sign without loss of generality), we have

$$\mathop{\mathbf{E}}_{\substack{y\sim\{0,1\}^{nk}\\ r_D\sim\{0,1\}^{t_D}}}[D(\mathsf{DP}_k(x,y);r_D)] - \mathop{\mathbf{E}}_{\substack{z\sim\{0,1\}^{nk+k}\\ r_D\sim\{0,1\}^{t_D}}}[D(z;r_D)] \geq \varepsilon.$$

Then by an averaging argument, we have

$$\Pr_{r_D}\left[\mathop{\mathbf{E}}_{y}[D(\mathsf{DP}_k(x,y);r_D)] - \mathop{\mathbf{E}}_{z}[D(z;r_D)] \geq \varepsilon/2\right] \geq \varepsilon/2. \tag{4}$$

In other words, with probability at least $\varepsilon/2$ over the internal randomness r_D, $D(-,r_D)$ is an $(\varepsilon/2)$-distinguisher for $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ and $\mathcal{U}_{nk+k}$. Let us say that r_D is *good* if this is true.

Let $\mathsf{Recon}^{(-)}\colon \{0,1\}^k \times \{0,1\}^r \to \{0,1\}^n$ be the reconstruction procedure in Lemma 14 that works for distinguishing parameter $\varepsilon/2$, where the randomness complexity r is $\mathsf{poly}(n/\varepsilon)$. We have

$$\begin{aligned}
&\Pr_{\substack{w\sim\{0,1\}^r\\ r_D\sim\{0,1\}^{t_D}}}\left[\exists\, \alpha \in \{0,1\}^k \text{ such that } \mathsf{Recon}^{D(-,r_D)}(\alpha,w) = x\right]\\
&\geq \Pr_{w,r_D}\left[\exists\, \alpha \in \{0,1\}^k \text{ such that } \mathsf{Recon}^{D(-,r_D)}(\alpha,w) = x \,\middle|\, r_D \text{ is good}\right]\cdot \Pr_{r_D}[r_D \text{ is good}]\\
&\geq \frac{1}{\mathsf{poly}(n/\varepsilon)}\cdot(\varepsilon/2). \qquad \text{(Lemma 14 and Equation (4))}
\end{aligned}$$

The above implies that for at least $1/\mathsf{poly}(n/\varepsilon)$ fraction of the randomness (w, r_D), there exists a program $\mathcal{M}$, which takes k bits for some α (that can depend on the randomness) and the advice of β used by D, such that $\mathcal{M}$ simulates $\mathsf{Recon}^{D(-,r_D)}$ on (α, w) and outputs x. Note that since Recon^D runs in time $t_D \cdot \mathsf{poly}(n/\varepsilon)$, our program $\mathcal{M}$ can be made to run in time $T := \widetilde{O}(t_D)\cdot\mathsf{poly}(n/\varepsilon)$. Therefore, we have

$$\mathsf{pK}^T_{1/\mathsf{poly}(n/\varepsilon)}(x \mid \beta) \leq k + O(\log(nt_D/\varepsilon)).$$

Then the lemma follows easily from Lemma 21. □

The following is an analog of Lemma 22 for the case of rK^t. In this case, we additionally assume the existence of a pseudodeterministic PRG (see Definition 8); later we will use Corollary 12 to argue the existence of a pseudodeterministic PRG from the assumption that $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$.

Lemma 23 (rK^t Reconstruction Lemma)**.** *Assume the existence of a pseudodeterministic PRG. For $\varepsilon > 0$, $x \in \{0,1\}^n$, $s \in \mathbb{N}$, and $k \in \mathbb{N}$ satisfying $k \leq 2n$, let D be a randomized algorithm that takes an advice β and runs in time t_D such that D ε-distinguishes $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Then there is a polynomial p'_{DP} such that*

$$\mathsf{rK}^{p'_{\mathsf{DP}}(nt_D/\varepsilon)}(x \mid \beta) \leq k + \log p'_{\mathsf{DP}}(nt_D/\varepsilon).$$

Proof. By averaging, arguing as in Lemma 22 up to Equation (4),

$$\Pr_{r_D \sim \{0,1\}^s}\left[\mathop{\mathbf{E}}_{y}[D(\mathsf{DP}_k(x,y), r_D)] - \mathop{\mathbf{E}}_{z}[D(z, r_D)] \geq \varepsilon/2\right] \geq \varepsilon/2. \tag{5}$$

That is, with probability at least $\varepsilon/2$ over its internal randomness r_D, $D(-, r_D)$ is an $(\varepsilon/2)$-distinguisher for $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ and $\mathcal{U}_{nk+k}$. Let us say that r_D is *good* if this is true.

Let $\mathsf{Recon}^{(-)}\colon \{0,1\}^k \times \{0,1\}^r \to \{0,1\}^n$ be the reconstruction procedure from Lemma 14 that works for distinguishing parameter $\varepsilon/2$. By definition of Recon, if r_D is good, then

$$\Pr_{w \sim \{0,1\}^r}\left[\mathsf{Recon}^{D(-,r_D)}(A(x,w), w) = x\right] \geq 1/\mathsf{poly}(n/\varepsilon),$$

and so

$$\Pr_{w, r_D}\left[\mathsf{Recon}^{D(-,r_D)}(A(x,w), w) = x\right] \geq \frac{\varepsilon}{2} \cdot \frac{1}{\mathsf{poly}(n/\varepsilon)} \overset{\text{def}}{=} \varepsilon'.$$

Observe that by Lemma 14, the condition $\mathsf{Recon}^{D(-,r_D)}(A(x,w), w) = x$ can be checked by a circuit of size $s = \mathsf{poly}(nt_D/\varepsilon)$ given (r_D, w) as input. Consider an assumed pseudodeterministic PRG

$$G : \{0,1\}^\ell \to \{0,1\}^{|r_D|+|w|}$$

that $(\varepsilon'/2)$-fools all circuits of size s, where $\ell \in O(\log(s/\varepsilon')) \subseteq O(\log(ns/\varepsilon))$. Let M_G be an algorithm as described in Definition 8, running in time $\mathsf{poly}(s/\varepsilon') \leq \mathsf{poly}(nt_D/\varepsilon)$, such that for all $\sigma \in \{0,1\}^\ell$,

$$\Pr_{r_M \sim \{0,1\}^{\mathsf{poly}(nt_D/\varepsilon)}}[M_G(\sigma, r_M) = G(\sigma)] \geq 1 - 2^{-s},$$

where r_M denotes the internal randomness of M_G. By definition of G,

$$\Pr_{\sigma}\left[\mathsf{Recon}^{D(-,G_0(\sigma))}(A(x, G_1(\sigma)), G_1(\sigma)) = x\right] \geq \varepsilon'/2,$$

where $G_0(\sigma) = G(\sigma)_{1\ldots|r_D|}$ (the $|r_D|$-length prefix) and $G_1(\sigma) = G(\sigma)_{|r_D|+1\ldots|r_D|+|w|}$ (the remaining suffix). So there must exist a seed $\sigma \in \{0,1\}^\ell$ such that

$$\mathsf{Recon}^{D(-,G_0(\sigma))}(A(x, G_1(\sigma)), G_1(\sigma)) = x.$$

Let σ be a seed with this property, and let $\alpha = A(x, G_1(\sigma)) \in \{0,1\}^k$. The definition of M_G implies that α and σ are such that

$$\begin{aligned}\Pr_{r_M}\left[\mathsf{Recon}^{D(-,M_{G_0}(\sigma,r_M))}(\alpha, M_{G_1}(\sigma, r_M)) = x\right] &\geq 1 - 2^{-s} \\ &> 2/3,\end{aligned}$$

where $M_{G_0}(\sigma, r_M)$ denotes the $|r_D|$-length prefix of $M_G(\sigma, r_M)$, and $M_{G_1}(\sigma, r_M)$ the remaining suffix of $M_G(\sigma, r_M)$. It is easy to verify that the reconstruction procedure $\mathsf{Recon}^{D(-,M_{G_0}(\sigma,r_M))}(\alpha, M_G(\sigma, -))$ runs in time $\mathsf{poly}(nt_D/\varepsilon)$ overall. By definition of rK^t, for some polynomial p'_{DP}, we get

$$\begin{aligned}\mathsf{rK}^{p'_{\mathsf{DP}}(nt_D/\varepsilon)}(x \mid \beta) &\le |\alpha| + |\sigma| + O(\log(ns/\varepsilon))\\ &\le k + \log p'_{\mathsf{DP}}(nt_D/\varepsilon),\end{aligned}$$

as required. □

3.3 Optimal source coding for pK^t

We will need the following result, which is an easy extension of an unconditional source coding theorem for pK^t described in [LOZ22].

Lemma 24 (Unconditional Source Coding for pK^t [LOZ22]). *There exists a polynomial p such that for any $T, a\colon \mathbb{N} \to \mathbb{N}$, $n \in \mathbb{N}$, $D_n \in \mathsf{Samp}[T(n)]/a(n)$, and $x \in \mathsf{Supp}(D_n)$,*

$$\mathsf{pK}^{p(T(n))}(x) \le \log(1/D_n(x)) + O(\log(T(n))) + a(n).$$

For a distribution D_n supported over X and an integer $m \ge 1$, we let $D_n^m := D_n \times \ldots \times D_n$ be the probability distribution supported over X^m obtained by taking the product of m independent copies of D_n.

As a consequence of Lemma 24, we obtain the following result.

Lemma 25 (Source Coding for m copies of D_n). *There exists a polynomial p such that for any $T, a\colon \mathbb{N} \to \mathbb{N}$, $n, m \in \mathbb{N}$, $D_n \in \mathsf{Samp}[T(n)]/a(n)$, and $x \in \mathsf{Supp}(D_n^m)$,*

$$\mathsf{pK}^{p(T(n),m)}(x) \le \log(1/D_n^m(x)) + O(\log(T(n))) + a(n) + O(\log(m)).$$

Proof. Since D_n can be sampled in time $T(n)$ using $a(n)$ bits of advice, it is possible to sample from D_n^m in time $\mathsf{poly}(T(n), m)$ using $a(n) + O(\log m)$ bits of advice, where the extra advice is used to encode m. The result now immediately follows from Lemma 24. □

3.4 Symmetry of information under randomized average-case easiness

Recently, [GK22], extending a technique from [Hir21a], have proved a symmetry of information result for the deterministic version of time-bounded Kolmogorov complexity K^t under the average-case easiness assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. We adapt and generalize their argument to prove an analogous result for conditional pK^t (and rK^t) under the randomized average-case easiness assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$ (respectively, $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$).

Lemma 26 (Symmetry of information for pK^t and rK^t under the average-case easiness of NP).

1. *If $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, then there exist polynomials p and p_0 such that for all sufficiently large $n, m \in \mathbb{N}$, all $t \ge p_0(n, m)$, and all $0 \le \tau \le t$ the following holds: for every $x \in \{0,1\}^n$ and every family $\{y_r \in \{0,1\}^m \mid r \in \{0,1\}^\tau\}$, with probability at least $9/10$ over $r \in \{0,1\}^\tau$,*

$$\mathsf{pK}^t(x, y_r \mid r) > \mathsf{pK}^{p(t)}(x \mid r) + \mathsf{pK}^{p(t)}(y_r \mid x, r) - \log p(t).$$

2. *If $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$, then there exist polynomials p' and p'_0 such that for all sufficiently large $n, m \in \mathbb{N}$, all $t \geq p'_0(n,m)$, and all $0 \leq \tau \leq t$ the following holds: for every $x \in \{0,1\}^n$ and every family $\{y_r \in \{0,1\}^m \mid r \in \{0,1\}^\tau\}$, with probability at least $9/10$ over $r \in \{0,1\}^\tau$,*

$$\mathsf{rK}^t(x, y_r \mid r) > \mathsf{rK}^{p'(t)}(x \mid r) + \mathsf{rK}^{p'(t)}(y_r \mid x, r) - \log p'(t).$$

Moreover, the special case without conditioning on r also holds for both items above, i.e., when r is the empty string (for $\tau = 0$) and y is an arbitrary single string.

Proof of Item 1. Define a language

$$L := \left\{(u, v, w, w', 1^s) \mid \exists \mathcal{M} \in \{0,1\}^s, \mathcal{M}(w, w') \text{ prints } uv \text{ in } |w| \text{ steps, and } s = |u| + |v| - 10\right\}.$$

Note that $L \in \mathsf{NP}$. Define a distribution family $D = \{D_{\langle nk+k, mk'+k', 2t, \tau, s\rangle}\}$ as follows: sample $u \sim \mathcal{U}_{nk+k}$, $v \sim \mathcal{U}_{mk'+k'}$, $w \sim \mathcal{U}_{2t}$, and $w' \sim \mathcal{U}_\tau$, and then output $(u, v, w, w', 1^s)$. Note that the ensemble D is in PSamp under an appropriate encoding of its defining tuple (for simplicity, we omit the padding in the output of D; see Section 2.1). Using the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, it follows that $(L, D) \in \mathsf{AvgBPP}$. Let B be a randomized algorithm for (L, D) as in Lemma 6.

Let $x \in \{0,1\}^n$ and $\{y_r\}_r \subseteq \{0,1\}^m$ be given (for sufficiently large n and m). Let $k, k' \in \mathbb{N}$ be arbitrary parameters such that $k \leq n$ and $k' \leq m$. Observe that there exists a polynomial p_0 such that for any $t \geq p_0(n,m)$, some constant d, and all $z \in \{0,1\}^{nk}$, $z' \in \{0,1\}^{mk'}$, and $r \in \{0,1\}^\tau$,

$$\mathsf{pK}^{2t}(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z') \mid r) \leq \mathsf{pK}^t(x, y_r \mid r) + |z| + |z'| + d \log t. \tag{6}$$

Here $p_0(n,m)$ reflects the time required to deterministically compute $(\mathsf{DP}_k(x;z), \mathsf{DP}_{k'}(y_r; z'))$ given x, y_r, z, z', and $d \log t$ bits of information to delineate x from y_r.

Let $t \geq p_0(n,m)$. By a counting argument, for independently random u, v, w, and w',

$$\Pr_{u,v,w,w'}\left[(u, v, w, w', 1^s) \in L\right] \leq \frac{2^s \cdot 2^{|w|+|w'|}}{2^{|u|+|v|+|w|+|w'|}} = 2^{s-|u|-|v|} = 2^{-10},$$

where the last equality is by the definition of L requiring that $s = |u| + |v| - 10$. Then

$$\begin{aligned}
&\Pr_{u,v,w,w',r_B}\left[B(u, v, w, w', 1^s) = 1\right] \\
&\leq \Pr_{u,v,w,w'}\left[(u, v, w, w', 1^s) \in L\right] + \Pr_{u,v,w,w',r_B}\left[B(u, v, w, w', 1^s) \neq L(u, v, w, w', 1^s)\right] \\
&\leq 2^{-10} + (3/n).
\end{aligned} \tag{7}$$

By Markov's inequality, for at least $9/10$ fraction of random strings $r \in \{0,1\}^\tau$, we get

$$\Pr_{u,v,w,r_B}\left[B(u, v, w, r, 1^s) = 1\right] \leq 10 \cdot \left(2^{-10} + (3/n)\right) \leq 1/10. \tag{8}$$

Fix any such r so that Equation (8) holds. This also fixes the string y_r.

Next we show, using a hybrid argument, that $B(-, \mathcal{U}_{2t}, r, 1^s)$ *cannot* distinguish between the uniform distribution and the distribution $(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z'))$, for random independent z, z', where $k \approx \mathsf{pK}^{p_{\mathsf{DP}}(t)}(x \mid r)$ and $k' \approx \mathsf{pK}^{p_{\mathsf{DP}}(t)}(y_r \mid x, r)$. This will imply that

$$\Pr_w\left[(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z'), w, r, 1^s) \in L\right] < 2/3$$

for some z, z', yielding the desired lower bound on $\mathsf{pK}^{2t}(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z') \mid r)$ via our choice of s and the definition of L. We give the details of the hybrid argument next.

First, toward a contradiction, suppose

$$\Pr_{z,v,w,r_B}[B(\mathsf{DP}_k(x; z), v, w, r, 1^s) = 1] > 1/2.$$

In this case, comparing with Equation (8), we get a randomized distinguisher (with advice) for $\mathsf{DP}_k(x; \mathcal{U}_{nk})$, defined by sampling $v \sim \mathcal{U}_{mk'+k'}$ and $w \sim \mathcal{U}_{2t}$, and outputting $B(-, v, w, r, 1^s)$. By Lemma 22,

$$\mathsf{pK}^{q(t)}(x \mid r) \leq k + \log q(t) \tag{9}$$

for some polynomial q such that $q(t) \geq p_{\mathsf{DP}}(t_B \cdot 3 \cdot n)$ whenever $t \geq n + m$ and $s \leq n^3$, where p_{DP} is the polynomial from Lemma 22 and t_B denotes the time required to compute $B(-, v, w, r, 1^s)$.

Define $k := \mathsf{pK}^{q(t)}(x \mid r) - \log q(t) - 1$ so that Equation (9) *does not hold.* Assume for now that $k > 0$. Hence,

$$\Pr_{z,v,w,r_B}[B(\mathsf{DP}_k(x; z), v, w, r, 1^s) = 1] \leq 1/2. \tag{10}$$

Again, toward a contradiction, suppose that for *all* z, z',

$$\Pr_{w}\left[(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z'), w, r, 1^s) \in L\right] \geq 2/3.$$

By definition of B, this implies that

$$\Pr_{z,z',w,r_B}\left[B(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z'), w, r, 1^s) = 1\right] \geq (2/3)(1 - 1/n) > 5/8.$$

In this case, comparing with Equation (10), we get a randomized distinguisher (with advice) B' for $\mathsf{DP}_{k'}(y_r; \mathcal{U}_{mk'})$, defined by sampling $z \sim \mathcal{U}_{nk}$, $w \sim \mathcal{U}_{2t}$, and outputting $B(\mathsf{DP}_k(x; z), -, w, r, 1^s)$. By Lemma 22,

$$\mathsf{pK}^{q'(t)}(y_r \mid x, r) \leq k' + \log q'(t) \tag{11}$$

for some polynomial q' with $q'(t) \geq p_{\mathsf{DP}}(t_{B'} \cdot 8 \cdot m)$ whenever $t \geq n + m$ and $s \leq n^3$, where $t_{B'}$ denotes the time required to compute B'.

We now choose $k' := \mathsf{pK}^{q'(t)}(y_r \mid x, r) - \log q'(t) - 1$ so that Equation (11) does not hold. Assume for now that $k' > 0$. Hence, there exist z and z' such that

$$\begin{aligned}&\Pr_{w}\left[(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z'), w, r, 1^s) \in L\right] \\ &= \Pr_{w}\left[\exists \mathcal{M} \in \{0,1\}^s, \mathcal{M}(w, r) \text{ outputs } (\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z')) \text{ within } |w| \text{ steps}\right] < 2/3,\end{aligned}$$

which implies that

$$\mathsf{pK}^{2t}(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z') \mid r) > s.$$

For this choice of z, z', by definition of s we get that $s = |u| + |v| - 10 = |z| + k + |z'| + k' - 10$, and so

$$\mathsf{pK}^{2t}(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z') \mid r) > |z| + k + |z'| + k' - 10.$$

Combining this inequality with Equation (6), we get

$$\begin{aligned}\mathsf{pK}^{t}(x, y_r \mid r) &\geq \mathsf{pK}^{2t}(\mathsf{DP}_k(x; z), \mathsf{DP}_{k'}(y_r; z') \mid r) - |z| - |z'| - d \log t \\ &> k + k' - d \log t - 10.\end{aligned} \tag{12}$$

The definitions of k and k' then imply that

$$\mathsf{pK}^t(x, y_r \mid r) > \mathsf{pK}^{q(t)}(x \mid r) + \mathsf{pK}^{q'(t)}(y_r \mid x, r) - \log q(t) - \log q'(t) - d \log t - 12.$$

For the polynomial $p(t) := (t^{d+1}) \cdot q(t) \cdot q'(t)$ and for every $t \geq p_0(|x|, |y|)$, we get

$$\mathsf{pK}^t(x, y_r \mid r) > \mathsf{pK}^{p(t)}(x \mid r) + \mathsf{pK}^{p(t)}(y_r \mid x, r) - \log p(t),$$

as desired.

Finally, consider the case that $k \leq 0$ or $k' \leq 0$. If $k \leq 0$, then $\mathsf{pK}^{q(t)}(x \mid r) \leq \log q(t) + 1$, implying that $\mathsf{pK}^{p(t)}(x \mid r) < \log p(t)$. But then the lemma simply follows from the fact that $\mathsf{pK}^t(x, y_r \mid r) \geq \mathsf{pK}^{p(t)}(y_r \mid x, r)$. Similarly, if $k' \leq 0$, then $\mathsf{pK}^{p(t)}(y_r \mid x, r) < \log p(t)$, and the lemma follows from the fact that $\mathsf{pK}^t(x, y_r \mid r) \geq \mathsf{pK}^{p(t)}(x \mid r)$. ☐

Proof of Item 2. Argue as in the proof of Item 1, but at Equation (9), apply Lemma 23 instead of Lemma 22 to get

$$\mathsf{rK}^{q(t)}(x \mid r) \leq k + \log q(t)$$

for some polynomial q such that $q(t) \geq p'_{\mathsf{DP}}(t_B \cdot 3 \cdot n)$, where p'_{DP} is the polynomial from Lemma 23 and t_B denotes the time required to compute $B(-, v, w, r, 1^s)$. We then define $k := \mathsf{rK}^{q(t)}(x \mid r) - \log q(t) - 1$ so the above equation does not hold. Similarly, at Equation (11), apply Lemma 23 to get

$$\mathsf{rK}^{q'(t)}(y_r \mid x, r) \leq k' + \log q'(t)$$

for some polynomial q' with $q'(t) \geq p'_{\mathsf{DP}}(t_{B'} \cdot 8 \cdot m)$, where $t_{B'}$ denotes the time required to compute the distinguisher B' defined as above. Then define $k' := \mathsf{rK}^{q'(t)}(y_r \mid x, r) - \log q'(t) - 1$.

Following the proof of Item 1 up to Equation (12), we get that

$$\mathsf{pK}^t(x, y_r \mid r) > k + k' - d \log t - \log n.$$

The definitions of k, k', and rK^t then imply that

$$\mathsf{rK}^t(x, y_r \mid r) > \mathsf{rK}^{q(t)}(x \mid r) + \mathsf{rK}^{q'(t)}(y_r \mid x, r) - \log q(t) - \log q'(t) - d \log t - 12,$$

and so

$$\mathsf{rK}^t(x, y_r \mid r) > \mathsf{rK}^{p(t)}(x \mid r) + \mathsf{rK}^{p(t)}(y_r \mid x, r) - \log p(t)$$

for the polynomial $p(t) := (t^{d+1}) \cdot q(t) \cdot q'(t)$. ☐

3.5 Approximating pK^t under randomized average-case easiness

Lemma 27. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, then there exists a polynomial* τ *such that the following promise problem is in* $\mathsf{promiseBPP}$*:*

$$\Pi_{\mathsf{YES}} := \left\{ (x, 1^s, 1^t) \mid \mathsf{pK}^t(x) \leq s \right\},$$
$$\Pi_{\mathsf{NO}} := \left\{ (x, 1^s, 1^t) \mid \mathsf{pK}^{\tau(t)}(x) > s + \log \tau(t) \right\}.$$

Proof. Let $k = s + \log n$. Consider the language

$$L := \{(\mathsf{DP}_k(x;z), w, 1^s, 1^n) \mid \exists \mathcal{M} \in \{0,1\}^s,\ \mathcal{M}(w) \text{ outputs } x \in \{0,1\}^n \text{ within } |w| \text{ steps}\}.$$

Note that $L \in \mathsf{NP}$. Define a distribution family $D = \{D_{\langle nk+k,t,s,n\rangle}\}$, each member of which does the following: sample $u \sim \mathcal{U}_{nk+k}$ and $w \sim \mathcal{U}_t$, and then output $(u, w, 1^s, 1^n)$. By assumption, $(L, D) \in \mathsf{AvgBPP}$. Let B be a randomized heuristic algorithm for (L, D) as described in Lemma 6.

Now, define an algorithm B':

> On input $(x, 1^s, 1^t)$ with $x \in \{0,1\}^n$, sample $z \sim \mathcal{U}_{nk}$ and $w \sim \mathcal{U}_t$, and then output $B(\mathsf{DP}_k(x;z), w, 1^s, 1^n)$.

Below, we show that B' solves $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ correctly with high probability in the worst case.

First, consider the case that $(x, 1^s, 1^t) \in \Pi_{\mathsf{YES}}$. By the definitions of L and pK, for any choice of $z \in \{0,1\}^{nk}$,

$$\Pr_w[(\mathsf{DP}_k(x;z), w, 1^s, 1^n) \in L] \geq 2/3.$$

The definition of B then implies that

$$\Pr_{w,z,r_B}[B(\mathsf{DP}_k(x;z), w, 1^s, 1^n) = 1] > 1/2,$$

and so

$$\Pr_{r_{B'}}\left[B'(x, 1^s, 1^t) = 1\right] > 1/2, \tag{13}$$

where r_B denotes the internal randomness of B, and $r_{B'} = (w, z, r_B)$ that of B'.

Now consider the case that $(x, 1^s, 1^t) \in \Pi_{\mathsf{NO}}$. For a contradiction, suppose that

$$\Pr_{w,z,r_B}[B(\mathsf{DP}_k(x;z), w, 1^s, 1^n) = 1] > 1/3. \tag{14}$$

By a counting argument, for randomly selected u and w,

$$\Pr_{u,w}[(u, w, 1^s, 1^n) \in L] \leq \frac{2^s \cdot 2^{nk} \cdot 2^{|w|}}{2^{nk+k+|w|}} = \frac{1}{n},$$

where the last line follows from the definition of $k = s + \log n$. Then by definition of B,

$$\Pr_{u,w,r_B}[B(u, w, 1^s, 1^n) = 1] \leq 4/n = o(1). \tag{15}$$

Comparing Equations (14) and (15), it is clear that $B(-, \mathcal{U}_t, 1^s, 1^n)$ (1/4)-distinguishes $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Lemma 22 implies that

$$\begin{aligned}\mathsf{pK}^{p'(t)}(x) &\leq k + \log p'(t)\\ &= s + \log n + \log p'(t),\end{aligned}$$

for some polynomial p' with $p'(t) \geq p_{\mathsf{DP}}(t_B \cdot 4 \cdot n)$, where p_{DP} is the polynomial from Lemma 22 and t_B denotes the time required to compute $B(-, \mathcal{U}_t, 1^s, 1^n)$. For the polynomial $\tau(t) = t \cdot p'(t)$, this means that $(x, 1^s, 1^t)$ is *not* in Π_{NO}, which gives the desired contradiction. By definition of B', we have that

$$\Pr_{r_{B'}}\left[B'(x, 1^s, 1^t) = 1\right] \leq 1/3. \tag{16}$$

By Equations (13) and (16), B' yields a $\mathsf{promiseBPP}$ algorithm for $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ via standard success amplification techniques. $\square$

Lemma 28 (Approximating pK^t)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, then there exist a polynomial* τ*, a constant* $C \geq 1$*, and a randomized algorithm* $\mathsf{Approx}_\tau\text{-}\mathsf{pK}$ *that, on input* $(x, 1^t)$ *where* $x \in \{0,1\}^n$ *and* $t \geq Cn$*, runs in time* $\mathsf{poly}(n,t)$ *and with probability at least* $1 - o(1)$ *outputs an integer* $\widetilde{s}$ *such that*

$$\mathsf{pK}^{\tau(t)}(x) - \log \tau(t) \ \leq \ \widetilde{s} \ \leq \ \mathsf{pK}^t(x).$$

Proof. Consider a polynomial-time randomized algorithm A that solves the promise problem from Lemma 27. Assume without loss of generality that on the inputs satisfying the promise its error is at most $1/n^2$, where $n = |x|$. Algorithm $\mathsf{Approx}_\tau\text{-}\mathsf{pK}$ runs A on $(x, 1^s, 1^t)$ for $s = 1, 2, \ldots, n + \log n$, and outputs the first $\widetilde{s}$ such that $A(x, 1^{\widetilde{s}}, 1^t) = 1$.

The correctness of $\mathsf{Approx}_\tau\text{-}\mathsf{pK}$ follows by a union bound. Indeed, if $s < \mathsf{pK}^{\tau(t)}(x) - \log \tau(t)$, i.e., $\mathsf{pK}^{\tau(t)}(x) > s + \log \tau(t)$, using the promise we get that $\Pr_A[A(x, 1^s, 1^t) = 1] \leq 1/n^2$. On the other hand, if $s = \mathsf{pK}^t(x)$, which implies that $\mathsf{pK}^t(x) \leq s$ and the promise is satisfied, we have $\Pr_A[A(x, 1^s, 1^t) = 1)] \geq 1 - 1/n^2$. Since $\mathsf{pK}^t(x) \leq n + \log n$ if $t \geq Cn$, where C is a sufficiently large constant, with high probability over the internal randomness of $\mathsf{Approx}_\tau\text{-}\mathsf{pK}$, it outputs a value $\widetilde{s}$ such that $\mathsf{pK}^{\tau(t)}(x) - \log \tau(t) \leq \widetilde{s} \ \leq \ \mathsf{pK}^t(x)$. ☐

3.6 Language compression under randomized average-case easiness

In some results that rely on language compression, it is useful to consider languages and promise problems that consist of strings of the form $(x, 1^\ell)$, where $|x| = \alpha(\ell)$ for some function α. More specifically, we need the following definition.

Definition 29 (Ensembles of Promise Problems[19])**.** Let $\alpha \colon \mathbb{N} \to \mathbb{N}$. We say that a promise problem $\Pi = (\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ is an *ensemble of promise problems with input size* α if

$$\Pi_{\mathsf{YES}} \cup \Pi_{\mathsf{NO}} \subseteq \left\{ (x, 1^\ell) \mid \ell \in \mathbb{N} \text{ and } |x| = \alpha(\ell) \right\}.$$

In this case, we let

$$\Pi_{\mathsf{YES},\ell} := \left\{ x \in \{0,1\}^{\alpha(\ell)} \mid (x, 1^\ell) \in \Pi_{\mathsf{YES}} \right\},$$

and similarly

$$\Pi_{\mathsf{NO},\ell} := \left\{ x \in \{0,1\}^{\alpha(\ell)} \mid (x, 1^\ell) \in \Pi_{\mathsf{NO}} \right\}.$$

Below we show a language compression result for problems in $\mathsf{promiseAM}$. Before stating and proving this result, note that it is not immediately clear what it means to have language compression for such a *promise* class. A reasonable definition may be that for every $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}}) \in \mathsf{promiseAM}$ and every $x \in \Pi_{\mathsf{YES}} \cap \{0,1\}^n$, we have $\mathsf{pK}^{\mathsf{poly}(n)}(x) \lesssim \log |\Pi_{\mathsf{YES}} \cap \{0,1\}^n|$. However, it is unclear how to show such a strong theorem. On the other hand, we manage to show a weaker version which instead says $\mathsf{pK}^{\mathsf{poly}(n)}(x) \lesssim \log \left| \overline{\Pi_{\mathsf{NO}}} \cap \{0,1\}^n \right|$ for every $x \in \Pi_{\mathsf{YES}} \cap \{0,1\}^n$, and it turns out that such a language compression for $\mathsf{promiseAM}$ suffices in some applications (see Section 4.6).

Theorem 30 (Language compression for pK^t under average-case easiness of NP)**.** *Let* $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}}) \in \mathsf{promiseAM}$ *be an ensemble of promise problems with input size* $\alpha \colon \mathbb{N} \to \mathbb{N}$*. Assume that* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*. Then there is a polynomial* p *such that for every* $\ell \in \mathbb{N}$ *and every* $x \in \Pi_{\mathsf{YES},\ell}$*,*

$$\mathsf{pK}^{p(\alpha(\ell)+\ell)}(x) \leq \log \left| \{0,1\}^{\alpha(\ell)} - \Pi_{\mathsf{NO},\ell} \right| + \log p(\alpha(\ell) + \ell).$$

[19] In the original definition of ensemble languages from [Hir21a], then length of x is not fixed and is required to be less than $\ell^{O(1)}$. For simplicity, here we require that this length is fixed according to ℓ.

Proof. Let $c > 0$ be a constant and V be a (deterministic) polynomial-time algorithm such that

$$(x, 1^\ell) \in \Pi_{\mathsf{YES}} \Rightarrow \Pr_{r \sim \{0,1\}^{(\alpha(\ell)+\ell)^c}} \left[\exists\, y \in \{0,1\}^{(\alpha(\ell)+\ell)^c}, V\Big(x, 1^\ell, y, r\Big) = 1\right] \geq 1 - 1/2^{\alpha(\ell)},$$

$$(x, 1^\ell) \in \Pi_{\mathsf{NO}} \Rightarrow \Pr_{r \sim \{0,1\}^{(\alpha(\ell)+\ell)^c}} \left[\forall\, y \in \{0,1\}^{(\alpha(\ell)+\ell)^c}, V\Big(x, 1^\ell, y, r\Big) = 0\right] \geq 1 - 1/2^{\alpha(\ell)},$$

Define the language

$$L' := \left\{ \Big(\mathsf{DP}_k(x; z), r, 1^\ell, 1^{\alpha(\ell)}\Big) \;\middle|\; |x| = \alpha(\ell), |z| = \alpha(\ell)k, |r| = (\alpha(\ell)+\ell)^c \text{ and } \exists\, y \in \{0,1\}^{(\alpha(\ell)+\ell)^c} \text{ such that } V\Big(x, 1^\ell, y, r\Big) = 1 \right\}.$$

Note that $L' \in \mathsf{NP}$. Define a distribution family $D = \left\{D_{\langle \alpha(\ell)k+k, (\alpha(\ell)+\ell)^c, \ell, \alpha(\ell)\rangle}\right\}$, each of which does the following: sample $u \sim \mathcal{U}_{\alpha(\ell)k+k}$, $r \sim \mathcal{U}_{(\alpha(\ell)+\ell)^c}$, and output $\left(u, r, 1^\ell, 1^{\alpha(\ell)}\right)$. By assumption, $(L', D) \in \mathsf{AvgBPP}$. Let B be a randomized heuristic algorithm for (L', D) as described in Lemma 6.

Consider any $\ell \in \mathbb{N}$ and any $x \in \Pi_{\mathsf{YES},\ell}$. Note that for any $z \in \{0,1\}^{\alpha(\ell)k}$,

$$\Pr_r \left[\Big(\mathsf{DP}_k(x; z), r, 1^\ell, 1^{\alpha(\ell)}\Big) \in L'\right] \geq 2/3.$$

Then by property of B,

$$\Pr_{z, r, r_B} \left[B\Big(\mathsf{DP}_k(x; z), r, 1^\ell, 1^{\alpha(\ell)}\Big) = 1\right] \geq 1/2. \tag{17}$$

On the other hand, for u and r selected uniformly at random, by a counting argument,

$$\begin{aligned} \Pr_{u,r}\left[\Big(u, r, 1^\ell, 1^{\alpha(\ell)}\Big) \in L'\right] &\leq \frac{2^{\alpha(\ell)k} \cdot 2^{(\alpha(\ell)+\ell)^c} \cdot \left(\left|\{0,1\}^{\alpha(\ell)} - \Pi_{\mathsf{NO},\ell}\right| + |\Pi_{\mathsf{NO},\ell}| / 2^{\alpha(\ell)}\right)}{2^{\alpha(\ell)k+k} \cdot 2^{(\alpha(\ell)+\ell)^c}} \\ &\leq 2^{-3}, \end{aligned}$$

where the last equality holds if we set $k := \log\left|\{0,1\}^{\alpha(\ell)} - \Pi_{\mathsf{NO},\ell}\right| + 4$. Then again by property of B,

$$\Pr_{u, r, r_B}\left[B\Big(u, r, 1^\ell, 1^{\alpha(\ell)}\Big) = 1\right] \leq 1/10.$$

Comparing with Equation (17), it is clear that $B\Big(-, \mathcal{U}_{(\alpha(\ell)+\ell)^c}, 1^\ell, 1^{\alpha(\ell)}\Big)$ is a randomized distinguisher for $\mathsf{DP}_k\big(x; \mathcal{U}_{\alpha(\ell)k}\big)$. Lemma 22 implies that

$$\begin{aligned} \mathsf{pK}^{\mathsf{poly}(\alpha(\ell)+\ell)}(x) &\leq k + O(\log(\alpha(\ell)+\ell)) \\ &\leq \log\left|\{0,1\}^{\alpha(\ell)} - \Pi_{\mathsf{NO},\ell}\right| + O(\log(\alpha(\ell)+\ell)), \end{aligned}$$

as desired. □

3.7 Probabilistic computational depth

The next lemma is a straightforward adaptation of an argument from [Hir21a]. We provide a proof for completeness.

Lemma 31 (Computational Depth Upper Bound)**.** *For all $x \in \{0,1\}^n$ and $r \in \{0,1\}^*$, and for all non-decreasing polynomials p_0 and p, if n is large enough then*

1. *there exists a time bound t_1 such that $p_0(n) \leq t_1 \leq 2^{n/\log n}$ and*
$$\mathsf{pK}^{t_1}(x \mid r) - \mathsf{pK}^{p(t_1)}(x \mid r) \leq O(n/\log n);$$
2. *there exists a time bound t_2 such that $p_0(n) \leq t_2 \leq 2^{n/\log n}$ and*
$$\mathsf{rK}^{t_2}(x \mid r) - \mathsf{rK}^{p(t_2)}(x \mid r) \leq O(n/\log n).$$

Proof. We will start by proving Item 1. Given $x \in \{0,1\}^n$ and polynomials p_0 and p, define the polynomial $\tau := p \circ p_0$. For an integer $I \geq 1$, consider the following telescoping sum:

$$\begin{aligned}\mathsf{pK}^{\tau(n)}(x \mid r) - \mathsf{pK}^{\tau^{I+1}(n)}(x \mid r) &= \left(\mathsf{pK}^{\tau(n)}(x \mid r) - \mathsf{pK}^{\tau^2(n)}(x \mid r)\right) \\ &\quad + \left(\mathsf{pK}^{\tau^2(n)}(x \mid r) - \mathsf{pK}^{\tau^3(n)}(x \mid r)\right) + \cdots + \left(\mathsf{pK}^{\tau^I(n)}(x \mid r) - \mathsf{pK}^{\tau^{I+1}(n)}(x \mid r)\right),\end{aligned}$$

where $\tau^i(-)$ denotes the composition of τ with itself i times. For any choice of x, p_0, and p as in the statement of the lemma, $\mathsf{pK}^{\tau(n)}(x \mid r) \leq n + d$, for some universal constant $d \geq 0$; hence, the above sum is at most $n + d$. By averaging, there is some index $i_0 \in [I]$ such that

$$\mathsf{pK}^{\tau^{i_0}(n)}(x \mid r) - \mathsf{pK}^{\tau^{i_0+1}(n)}(x \mid r) \leq \frac{n+d}{I}.$$

For this i_0, define $t_1 := \tau^{i_0}(n)$. Note that $t_1 \geq \tau(n) \geq p_0(n)$, since $i_0 \geq 1$ and $p(\ell) \geq \ell$ for every input ℓ. Letting $c \in \mathbb{N}$ be such that $\tau(n) \leq n^c$ for sufficiently large n, define $I := \log_c(n/(\log_2 n)^2)$. Then $t_1 \leq n^{c^I} = 2^{n/\log n}$. Moreover,

$$\begin{aligned}\mathsf{pK}^{t_1}(x \mid r) - \mathsf{pK}^{p(t_1)}(x \mid r) &\leq \mathsf{pK}^{t_1}(x \mid r) - \mathsf{pK}^{\tau(t_1)}(x \mid r) \\ &\leq O(n/\log n),\end{aligned}$$

as desired.

The proof of Item 2 is similar. □

4 Probabilistic Worst-Case to Average-Case Reductions

4.1 Auxiliary lemmas

Given a language $L \in \mathsf{NP}$ with a corresponding verifier V, and $x \in L$, let y_x be the lexicographically first witness that $x \in L$. It is not hard to compute y_x from x if an NP oracle is available, by performing a standard search-to-decision reduction. For this reason, the time-bounded Kolmogorov complexity of the pair (x, y_x) is essentially that of x, in the presence of an NP oracle. We will show, more generally, that the oracle can be eliminated under an appropriate average-case easiness assumption, even for the following randomized versions of Σ_ℓ^{P}, denoted $\mathsf{BP}_\delta \circ \Sigma_\ell^{\mathsf{P}}$.

Definition 32 ($\mathsf{BP}_\delta \circ \Sigma_\ell^{\mathsf{P}}$)**.** A language L is in the class $\mathsf{BP}_\delta \circ \Sigma_\ell^{\mathsf{P}}$ if there is a $\mathsf{BP} \circ \Sigma_\ell \circ \mathsf{P}$ formula

$$\phi(x) = \mathsf{BP}r\ \exists y_1\ \forall y_2\ \ldots\ Qy_\ell\ \ R(x, y_1, y_2, \ldots, y_\ell, r)$$

for a polytime predicate R and all string variables $y_1, \ldots, y_\ell, r$ of length $\mathsf{poly}(|x|)$, such that, for all $x \in \{0,1\}^n$,

$$\begin{aligned} x \in L &\implies \Pr_r\left[\exists\, y_1\ \forall\, y_2\ \ldots\ Q\, y_\ell\ \ R(x, y_1, y_2, \ldots, y_\ell, r)\right] \geq \delta, \text{ and} \\ x \notin L &\implies \Pr_r\left[\forall\, y_1\ \exists\, y_2\ \ldots\ \bar{Q}\, y_\ell\ \ \neg R(x, y_1, y_2, \ldots, y_\ell, r)\right] \geq \delta. \end{aligned}$$

Lemma 33 (Oracle Elimination)**.** *For any $\ell \geq 1$, suppose $\mathsf{Dist}\Sigma_{\ell+1}^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. For an arbitrary $L \in \mathsf{BP}_\delta \circ \Sigma_\ell^{\mathsf{P}}$, let $x \in L_n$ be sufficiently large, let $r \in \{0,1\}^{n^c}$ be any random string under the* BP *quantifier such that there exists a witness for the left-most $\exists$ quantifier in the definition of L, and let $y_{x,r} \in \{0,1\}^{n^c}$ be the lexicographically first such L-witness for x and randomness r, for some constant $c > 0$.*

1. *There exist polynomials q and q_0 such that, for all sufficiently large $n \in \mathbb{N}$, all $x \in L_n$, and all $t \geq q_0(n)$, we have with probability at least $\delta - 1/10$ over $r \in \{0,1\}^{n^c}$ that an L-witness $y_{x,r}$ exists, and*
$$\mathsf{pK}^{q(t)}(x, y_{x,r} \mid r) \leq \mathsf{pK}^t(x \mid r) + \log q(t).$$

2. *Assume in addition that $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$ (which holds in particular for $\ell \geq 2$). Then there exist polynomials q' and q_0' such that, for all sufficiently large $n \in \mathbb{N}$, all $x \in L_n$, and all $t \geq q_0'(n)$, we have with probability at least $\delta - 1/10$ over $r \in \{0,1\}^{n^c}$ that an L-witness $y_{x,r}$ exists, and*
$$\mathsf{rK}^{q'(t)}(x, y_{x,r} \mid r) \leq \mathsf{rK}^t(x \mid r) + \log q'(t).$$

Moreover, for the case of no BP *quantifier, i.e., for $L \in \Sigma_\ell^{\mathsf{P}}$, we get the same conclusions (with probability 1) without conditioning on r.*

Proof of Item 1. Define

$$L' := \left\{ \left(\mathsf{DP}_k(x, y; z), w, w', 1^k, 1^s, 1^n\right) \;\middle|\; \begin{array}{l} \exists\, \mathcal{M} \in \{0,1\}^s, \mathcal{M}^{\Sigma_\ell^{\mathsf{P}}}(w, w') \text{ prints } (x, y) \in \{0,1\}^{n+n^c} \\ \text{within } |w| \text{ steps, where } |w'| = n^c, \text{ and } s = k - 10 \end{array} \right\}.$$

More formally, in this and subsequent proofs, the above notation should be interpreted as follows: a string $(u, w, w', 1^k, 1^s, 1^n)$ belongs to L' iff there exist some $(x, y) \in \{0,1\}^{n+n^c}$ and $\mathcal{M} \in \{0,1\}^s$ such that $\mathsf{DP}_k(x, y; z) = u$, where $u = (z, \alpha)$ for some $z \in \{0,1\}^{(n+n^c)k}$ and $\alpha \in \{0,1\}^k$, and $\mathcal{M}^{\Sigma_\ell^{\mathsf{P}}}(w, w')$ outputs (x, y) within $|w|$ steps, for $|w'| = n^c$.

Note that $L' \in \mathsf{NP}^{\Sigma_\ell^{\mathsf{P}}}$. Define a distribution family $D = \{D_{\langle (n+n^c)k+k, 2t, n^c, k, s, n\rangle}\}$ as follows: sample $u \sim \mathcal{U}_{(n+n^c)k+k}$, $w \sim \mathcal{U}_{2t}$, $w' \sim \mathcal{U}_{n^c}$, and output $(u, w, w', 1^k, 1^s, 1^n)$, for $s = k - 10$. By assumption, $(L', D) \in \mathsf{AvgBPP}$. Let B be a randomized algorithm for (L', D) as in Lemma 6.

For all (sufficiently large) n, k, and s, and for independent uniformly random u, w, and w',

$$\begin{aligned}\mathop{\mathbf{Pr}}_{u,w,w'}\left[(u,w,w',1^k,1^s,1^n)\in L'\right] &= \mathop{\mathbf{Pr}}_{u,w,w'}\Big[\exists\,(x,y)\in\{0,1\}^{n+n^c}, \exists\, z\in\{0,1\}^{(n+n^c)k}, \exists\,\mathcal{M}\in\{0,1\}^s,\\ &\qquad\qquad \mathcal{M}^{\Sigma_\ell^{\mathsf{P}}}(w,w')=(x,y)\ \wedge\ u=\mathsf{DP}_k(x,y;z)\Big]\\ &\le \frac{2^{s+|w|+|w'|+|z|}}{2^{k+|w|+|w'|+|z|}} = 2^{s-k} = 2^{-10},\end{aligned}$$

where the inequality is by a union bound and a counting argument, and the last equality by the condition that $s=k-10$ in the definition of L'. Hence,

$$\begin{aligned}&\mathop{\mathbf{Pr}}_{u,w,w',r_B}\left[B(u,w,w',1^k,1^s,1^n)=1\right]\\ &\le \mathop{\mathbf{Pr}}_{u,w,w'}\left[(u,w,w',1^k,1^s,1^n)\in L'\right] + \mathop{\mathbf{Pr}}_{u,w,w',r_B}\left[B(u,w,w',1^k,1^s,1^n)\neq L'(u,w,w',1^k,1^s,1^n)\right]\\ &\le 2^{-10}+(3/n).\end{aligned}$$

By Markov's inequality, for at least $9/10$ fraction of strings r,

$$\mathop{\mathbf{Pr}}_{u,w,r_B}\left[B(u,w,r,1^k,1^s,1^n)=1\right] \le 10\cdot(2^{-10}+(3/n)) \le 1/10. \tag{18}$$

Let $x\in L_n$ be arbitrary. By definition of the BP_δ quantifier, for at least δ fraction of random strings $r\in\{0,1\}^{n^c}$, there exists an L-witness, and hence, the lexicographically first L-witness $y_{x,r}$. By the above, for at least $\delta-1/10$ of random r, we have that both a witness $y_{x,r}$ exists and Equation (18) holds. Fix any such r.

Observe that for some polynomial q_0 (dependent on L) and some constant d, for any $t\ge q_0(n)$,

$$\mathsf{pK}^{2t,\Sigma_\ell^{\mathsf{P}}}(x,y_{x,r}\mid r)\le \mathsf{pK}^t(x\mid r)+d\log n =: s(x,r). \tag{19}$$

In particular, $q_0(n)$ reflects the time required to deterministically compute $y_{x,r}$, given x and r, by a search-to-decision procedure for L using a Σ_ℓ^{P}-oracle.

Define $s:=s(x,r)$ (which determines $k=s+10$), and let $t\ge q_0(n)$. By the definitions of $\mathsf{pK}^{2t,\Sigma_\ell^{\mathsf{P}}}$ and L', for every $z\in\{0,1\}^{(n+n^c)k}$,

$$\mathop{\mathbf{Pr}}_{w}\left[(\mathsf{DP}_k(x,y_{x,r};z),w,r,1^k,1^s,1^n)\in L'\right]\ge 2/3.$$

Then by the definition of B,

$$\begin{aligned}&\mathop{\mathbf{Pr}}_{z,w,r_B}\left[B(\mathsf{DP}_k(x,y_{x,r};z),w,r,1^k,1^s,1^n)=1\right]\\ &\ge \mathop{\mathbf{Pr}}_{z,w}\left[(\mathsf{DP}_k(x,y_{x,r};z),w,r,1^k,1^s,1^n)\in L'\right]\cdot \mathop{\mathbf{Pr}}_{r_B}\left[B(\omega)=L'(\omega)\mid \omega\in L'\right]\\ &\ge \frac{2}{3}\cdot(1-(1/n))\\ &\ge 2/3-o(1).\end{aligned}\tag{20}$$

Comparing Equation (18) with Equation (20), it is clear that $B(-, \mathcal{U}_{2t}, r, 1^k, 1^s, 1^n)$ is a randomized algorithm (with advice) that (1/2)-distinguishes $\mathsf{DP}_k(x, y_{x,r}; \mathcal{U}_{(n+n^c)k})$ from uniform. Lemma 22 implies that

$$\begin{aligned} \mathsf{pK}^{p'(t)}(x, y_{x,r} \mid r, s, n, t) &\leq k + \log p'(t) \\ &= s + 10 + \log p'(t) \\ &= \mathsf{pK}^t(x \mid r) + d \log n + \log p'(t) + 10, \end{aligned} \tag{21}$$

for some polynomial p' with $p'(t) \geq p_{\mathsf{DP}}(t_B \cdot 3 \cdot (n + n^c))$, where p_{DP} is the polynomial from Lemma 22 and t_B denotes the time required to compute $B(-, \mathcal{U}_{2t}, r, 1^k, 1^s, 1^n)$. As the advice (s, n, t) can be encoded with $\log s + \log n + \log t + O(\log \log n) \leq (3.1) \cdot \log t$ bits, it follows that $\mathsf{pK}^{q(t)}(x, y_{x,r} \mid r) \leq \mathsf{pK}^t(x \mid r) + \log q(t)$ for the polynomial $q(t) := t^{(d+4)} \cdot p'(t)$.

The "Moreover" statement follows the same argument, dropping any mention of w' (and r). $\square$

Proof of Item 2. The proof is similar to that of Item 1, except at Equation (19), we observe that for all $t \geq q_0(n)$,

$$\begin{aligned} \mathsf{pK}^{2t, \Sigma_\ell^{\mathsf{P}}}(x, y_{x,r} \mid r) &\leq \mathsf{pK}^t(x \mid r) + d \log n \\ &\leq \mathsf{rK}^t(x \mid r) + d \log n, \end{aligned}$$

and then define $s(x, r) := \mathsf{rK}^t(x \mid r) + d \log n$. We then follow the proof of Item 1 up to Equation (21), where we apply Lemma 23 instead of Lemma 22 to obtain

$$\mathsf{rK}^{p''(t)}(x, y_{x,r} \mid r, s, n, t) \leq \mathsf{rK}^t(x \mid r) + d \log n + \log p''(t) + 10$$

for some polynomial p'' with $p''(t) \geq p'_{\mathsf{DP}}(t_B \cdot 3 \cdot (n + n^c))$, where p'_{DP} is the polynomial from Lemma 23 and t_B denotes the time required to compute $B(-, \mathcal{U}_{2t}, r, 1^k, 1^s, 1^n)$. It follows that $\mathsf{rK}^{q'(t)}(x, y_{x,r} \mid r) \leq \mathsf{rK}^t(x \mid r) + \log q'(t)$ for the polynomial $q'(t) := t^{(d+4)} \cdot p''(t)$. $\square$

Lemma 34 (Witness Compression)**.** *For any $\ell \geq 1$, suppose $\mathsf{Dist}\Sigma_{\ell+1}^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. For an arbitrary $L \in \mathsf{BP}_\delta \circ \Sigma_\ell^{\mathsf{P}}$, let $x \in L_n$ be sufficiently large, $r \in \{0,1\}^{n^c}$ be any good random string, and let $y_{x,r} \in \{0,1\}^{n^c}$ be the lexicographically first L-witness for x with respect to randomness r, for some constant $c > 0$.*

1. *Let p_0, p and q_0, q be the polynomials from Items 1 of Lemmas 26 and 33 respectively. Then for every $t \geq \max\{p_0(n), q_0(n + n^c)\}$, with probability at least $\delta - 1/5$ over r, an L-witness $y_{x,r}$ exists, and*

$$\mathsf{pK}^{p(q(t))}(y_{x,r} \mid x, r) \leq \Big(\mathsf{pK}^t(x \mid r) - \mathsf{pK}^{p(q(t))}(x \mid r)\Big) + 2 \log p(q(t)).$$

2. *Assume in addition that $\mathsf{DistP}^{\Sigma_2^{\mathsf{P}}} \subseteq \mathsf{AvgBPP}$ (which holds in particular when $\ell \geq 2$). Let p'_0, p' and q'_0, q' be the polynomials from Items 2 of Lemmas 26 and 33 respectively. Then for every $t \geq \max\{p'_0(n), q'_0(n + n^c)\}$, with probability at least $\delta - 1/5$ over r, an L-witness $y_{x,r}$ exists, and*

$$\mathsf{rK}^{p'(q'(t))}(y_{x,r} \mid x, r) \leq \Big(\mathsf{rK}^t(x \mid r) - \mathsf{rK}^{p'(q'(t))}(x \mid r)\Big) + 2 \log p'(q'(t)).$$

Moreover, for $L \in \Sigma_\ell^{\mathsf{P}}$, we get the same conclusions without conditioning on r.

Proof. We start by proving Item 1. By Item 1 of Lemma 33 and Lemma 26, we get by a union bound that, with probability at least $\delta - 2/10$ over r,

$$\mathsf{pK}^{q(t)}(x, y_{x,r} \mid r) \leq \mathsf{pK}^{t}(x \mid r) + \log q(t),$$

and

$$\mathsf{pK}^{q(t)}(x, y_{x,r} \mid r) > \mathsf{pK}^{p(q(t))}(x \mid r) + \mathsf{pK}^{p(q(t))}(y_{x,r} \mid x, r) - \log p(q(t)).$$

Combining the previous two inequalities,

$$\begin{aligned}\mathsf{pK}^{p(q(t))}(y_{x,r} \mid x, r) &< \mathsf{pK}^{q(t)}(x, y_{x,r} \mid r) - \mathsf{pK}^{p(q(t))}(x \mid r) + \log p(q(t)) \\ &\leq \left(\mathsf{pK}^{t}(x \mid r) - \mathsf{pK}^{p(q(t))}(x \mid r)\right) + 2\log p(q(t)).\end{aligned}$$

The "Moreover" part follows by applying the "no random r" versions of Lemma 33 and Lemma 26. The proof of Item 2 is the same, except we apply Items 2 of Lemmas 33 and 26 respectively. ☐

We will also need the following analogue of Lemma 33 for the case of unique witnesses.

Lemma 35 (Oracle Elimination for UP)**.** *Suppose* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, and let* $L \in \mathsf{UP}$*. Let* $x \in L_n$ *be sufficiently large, and let* $y_x \in \{0,1\}^{n^c}$ *be the unique* L*-witness for* x*, for some constant* $c > 0$*. There exists a polynomial* q *such that for all* $t \geq n + n^c$*,*

$$\mathsf{pK}^{q(t)}(x, y_x) \leq \mathsf{pK}^{t}(x) + \log q(t).$$

Proof. Let $L \in \mathsf{UP}$ with deterministic verifier V running in time n^c on inputs $x \in \{0,1\}^n$ and $y \in \{0,1\}^{n^c}$. Let

$$L' := \Big\{ (\mathsf{DP}_k(x, y; z), w, 1^s, 1^n) \mid \exists \mathcal{M} \in \{0,1\}^s, \mathcal{M}(w) \text{ outputs } x \in \{0,1\}^n \text{ within } |w| \text{ steps,} \\ \text{where } k = s + 10, \text{ and } V(x, y) = 1 \Big\}.$$

Note that $L' \in \mathsf{NP}$. Define a distribution family $D = \{D_{\langle (n+n^c)k+k,t,s,n\rangle}\}$ as follows: sample $u \sim \mathcal{U}_{(n+n^c)k+k}$, $w \sim \mathcal{U}_t$, and output $(u, w, 1^s, 1^n)$. By assumption, $(L', D) \in \mathsf{AvgBPP}$. Let B be a randomized heuristic algorithm for (L', D) as described in Lemma 6.

Let $x \in L_n$ be sufficiently large, and let $y_x \in \{0,1\}^{n^c}$ be the *unique* L-witness for x. Define $s := \mathsf{pK}^t(x)$, and let $t \geq n + n^c$. By definition of pK^t and L', for any $z \in \{0,1\}^{(n+n^c)k}$,

$$\Pr_{w}\left[(\mathsf{DP}_k(x, y_x; z), w, 1^s, 1^n) \in L'\right] \geq 2/3.$$

Then by definition of B,

$$\begin{aligned}&\Pr_{z,w,r_B}\left[B(\mathsf{DP}_k(x, y_x; z), w, 1^s, 1^n) = 1\right] \\ &\geq \Pr_{z,w}\left[(\mathsf{DP}_k(x, y_x; z), w, 1^s, 1^n) \in L'\right] \cdot \Pr_{r_B}\left[B(\omega) = L'(\omega) \mid \omega \in L'\right] \\ &\geq \frac{2}{3} \cdot (1 - (1/n)) \geq 1/2. \qquad (22)\end{aligned}$$

On the other hand, for u and w selected uniformly at random,

$$\begin{aligned}\Pr_{u,w}\left[(u,w,1^s,1^n)\in L'\right] &= \Pr_{u,w}\Big[\exists\,(x,y)\in\{0,1\}^{n+n^c}, \exists\, z\in\{0,1\}^{(n+n^c)k}, \exists\,\mathcal{M}\in\{0,1\}^s,\\ &\qquad\qquad \mathcal{M}(w)=x \,\wedge\, V(x,y)=1 \,\wedge\, u=\mathsf{DP}_k(x,y;z)\Big]\\ &\le \frac{2^{s+|w|+|z|}}{2^{k+|w|+|z|}} = 2^{-10},\end{aligned}$$

where the inequality is by a union bound and a counting argument, and the last equality by definition of $k=s+10=\mathsf{pK}^t(x)+10$. Then

$$\begin{aligned}\Pr_{u,w,r_B}\left[B(u,w,1^s,1^n)=1\right] &\le \Pr_{u,w}\left[(u,w,1^s,1^n)\in L'\right]+\Pr_{u,w,r_B}\left[B(u,w,1^s,1^n)\neq L'(u,w,1^s,1^n)\right]\\ &\le 2^{-10}+(3/n)\le 1/10.\end{aligned}$$

Comparing with Equation (22), it is clear that $B(-,\mathcal{U}_t,1^s,1^n)$ is a randomized distinguisher for $\mathsf{DP}_k\big(x,y_x;\mathcal{U}_{(n+n^c)k}\big)$. Lemma 22 implies that

$$\begin{aligned}\mathsf{pK}^{p'(t)}(x,y_x) &\le k+\log p'(t)\\ &= \mathsf{pK}^t(x)+10+\log p'(t)\end{aligned}$$

for some polynomial p' with $p'(t)\ge p_{\mathsf{DP}}(t_b\cdot 3\cdot(n+n^c))$, where p_{DP} is the polynomial from Lemma 22 and t_B denotes the time required to compute $B(-,\mathcal{U}_t,1^s,1^n)$. It follows that $\mathsf{pK}^{q(t)}(x,y_x)\le \mathsf{pK}^t(x)+\log q(t)$ for the polynomial $q(t):=t\cdot p'(t)$. $\square$

4.2 Case of AM

Theorem 36. *If* $\mathsf{Dist}\Sigma_2^{\mathsf{P}}\subseteq\mathsf{AvgBPP}$*, then* $\mathsf{AM}\subseteq\mathsf{BPTIME}\left[2^{O(n/\log n)}\right]$.

Proof. Consider an arbitrary $L\in\mathsf{AM}$ given by a $\mathsf{BP}\circ\Sigma_1^{\mathsf{P}}$ formula such that, for every $x\in\{0,1\}^n$,

$$\begin{aligned} x\in L &\implies \mathsf{BP}_{(1-2^{-n})}r\ \exists y\ V(x,y,r),\\ x\notin L &\implies \mathsf{BP}_{(1-2^{-n})}r\ \forall y\ \neg V(x,y,r),\end{aligned}$$

for a polytime verifier $V(x,y,r)$, where $|r|,|y|\le n^c$ for some constant $c>0$, and the notation $\mathsf{BP}_{(\delta)}r\ \phi(-,r)$, for a formula ϕ, means $\mathbf{Pr}_r[\phi(-,r)]\ge\delta$.

For a given (sufficiently large) $x\in L_n$, and a good random string r, let $y_{x,r}$ be the lexicographically first L-witness for x,r. By Lemma 34, for some constant $a>0$, and for all large enough time bounds t, we have, with probability at least $1-2^{-n}-1/5$ over random strings r, that

$$\mathsf{pK}^{t^a}(y_{x,r}\mid x,r)\le\big(\mathsf{pK}^t(x\mid r)-\mathsf{pK}^{t^a}(x\mid r)\big)+(2a)\log t.$$

By Lemma 31, there exists a time bound $t\le 2^{O(n/\log n)}$ such that $\mathsf{pK}^t(x\mid r)-\mathsf{pK}^{t^a}(x\mid r)\le bn/\log n$, for some universal constant $b>0$. Hence, for some universal constant $d>0$ (independent of x,r),

$$\mathsf{pK}^{2^{dn/\log n}}(y_{x,r}\mid x,r)\le dn/\log n.$$

A randomized algorithm for L is now obvious:

Given $x \in \{0,1\}^n$, sample uniformly random strings $r \in \{0,1\}^{|x|^c}$ and $w \in \{0,1\}^{2^{dn/\log n}}$, and then exhaustively search over all $dn/\log n$-bit machines $\mathcal{M}$, running each $\mathcal{M}(w,x,r)$ for $2^{dn/\log n}$ steps to produce a candidate witness y, accepting if $V(x,y,r)$ accepts.

The runtime $2^{O(n/\log n)}$ is clear. For correctness, for every $x \in L_n$, we know there is at least a $(4/5-2^{-n})$ fraction of good random strings r for which a witness $y_{x,r}$ will be found by our algorithm, with probability at least $2/3$ over random w's (by the definition of pK). So our algorithm accepts $x \in L$ with probability at least $(2/3)(4/5) - o(1) > 1/2$. On the other hand, for any $x \notin L$, our algorithm may find a witness y and accept x for at most 2^{-n} fraction of r's. □

4.3 Case of UP

Theorem 37. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{UP} \subseteq \mathsf{RTIME}\left[2^{O(n/\log n)}\right]$.

Proof. Let $L \in \mathsf{UP}$, $x \in L_n$ for n sufficiently large, and $y_x \in \{0,1\}^{n^c}$ the unique L-witness for x, for some constant $c > 0$. Let V be a deterministic verifier for L running in time n^c. Let p_0, p, and q be the polynomials from Lemmas 26 (Item 1) and 35 respectively.

By Lemma 31, there exists $p_0(n+n^c) \le t \le 2^{n/\log n}$ such that

$$\mathsf{pK}^t(x) - \mathsf{pK}^{p(q(t))}(x) \le O(n/\log n).$$

For such a t, combining Lemma 26 and Lemma 35 as in the proof of Lemma 34, we obtain

$$\mathsf{pK}^{p(q(t))}(y_x \mid x) \le \left(\mathsf{pK}^t(x) - \mathsf{pK}^{p(q(t))}(x)\right) + 2\log p(q(t)).$$

The above inequalities imply that for some constant $d \in \mathbb{N}$,

$$\mathsf{pK}^{2^{dn/\log n}}(y_x \mid x) \le dn/\log n.$$

We are now ready to define a probabilistic algorithm A deciding L.

On input $x \in \{0,1\}^n$, A samples $w \sim \{0,1\}^{2^{dn/\log n}}$. It then exhaustively searches over all machines $\mathcal{M} \in \{0,1\}^{dn/\log n}$, running $\mathcal{M}(w,x)$ for $2^{dn/\log n}$ steps to produce some output y. A accepts iff a y is obtained such that $V(x,y) = 1$.

If $x \in L$, then A accepts with probability at least $2/3$ over w; if $x \notin L$, then A never accepts. □

4.4 Case of PH

As a warm-up, we consider a special case of PH languages L, where a Σ_ℓ^{P} verifier for $L \in \Sigma_\ell^{\mathsf{P}}$ has all its $\exists/\forall$ quantifiers over binary strings of length $O(|x|)$ for a given input x. Denote the class of such Σ_ℓ^{P} languages by $\mathsf{lin}\Sigma_\ell^{\mathsf{P}}$.

Theorem 38. *For an arbitrary $\ell \ge 1$, suppose that* $\mathsf{Dist}\Sigma_{\ell+1}^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$*. Then*

$$\mathsf{lin}\Sigma_\ell^{\mathsf{P}} \subseteq \mathsf{BPTIME}\left[2^{O(n/\log n)}\right].$$

Proof. The proof is by induction on ℓ. The base case of $\ell = 1$ follows from Theorem 36. We will argue the case of $\ell \geq 2$. Consider an arbitrary language $L \in \mathsf{lin}\Sigma_\ell^{\mathsf{P}}$. By definition, there is a polytime predicate R and a constant $a > 0$ such that, for every $x \in \{0,1\}^n$,

$$x \in L \iff \exists\, y_1\, \forall\, y_2 \,\ldots\, Q\, y_\ell \;\; R(x, y_1, y_2, \ldots, y_\ell),$$

where $|y_i| \leq an$ for all $1 \leq i \leq \ell$. Define the language

$$L' = \{(x, y) \mid \forall\, y_2 \,\ldots\, Q\, y_\ell \;\; R(x, y, y_2, \ldots, y_\ell)\}\,.$$

Note that $L' \in \mathsf{lin}\Pi_{\ell-1}^{\mathsf{P}}$, and so, by the Inductive Hypothesis, $L' \in \mathsf{BPTIME}[2^{O(n/\log n)}]$, since $|(x, y)| \leq O(|x|)$ by assumption. By standard success amplification, we may assume that this probabilistic algorithm for L' has error probability at most 2^{-n}.

On the other hand, by Items 1 of Lemmas 34 and 31, we have that, for some universal constant $d > 0$, for every sufficiently large input $x \in L_n$,

$$\mathsf{pK}^{2^{dn/\log n}}(y_x \mid x) \leq dn/\log n,$$

where y_x is the lexicographically first L-witness for $x \in L$.

We are now ready to define a probabilistic algorithm A deciding L.

> On input $x \in \{0,1\}^n$, A samples $w \sim \{0,1\}^{2^{dn/\log n}}$. It then exhaustively searches over all machines $\mathcal{M} \in \{0,1\}^{dn/\log n}$, running $\mathcal{M}(w, x)$ for $2^{dn/\log n}$ steps to produce some output y. A accepts iff a y is obtained such that $(x, y) \in L'$, where the latter is checked using a $\mathsf{BPTIME}[2^{O(n/\log n)}]$ algorithm for L' shown to exist earlier.

The overall runtime $2^{O(n/\log n)}$ of the described algorithm A is clear. For correctness, consider an arbitrary $x \in L_n$. With probability at least $2/3$ over w's, A will check if $(x, y_x) \in L'$. The latter check will succeed with probability at least $1 - 2^{-n}$, by assumption. Therefore, A will correctly accept x with probability at least $(2/3)(1 - 2^{-n}) \geq 6/10$. Next consider an arbitrary $x \in \{0,1\}^n$ such that $x \notin L$. For this x, no witnesses exist, and hence every string y tried by A is such that $(x, y) \notin L'$. The probability that at least one such y is incorrectly accepted by A is, by the union bound, at most $2^{dn/\log n} \cdot 2^{-n} \leq 2^{-n/2}$, for all sufficiently large n. $\square$

It is observed in [GK22, Corollary 17] that, for every $\ell \geq 1$, $\Sigma_\ell^{\mathsf{P}} = \mathsf{lin}\Sigma_\ell^{\mathsf{P}}$, under the assumption that $\mathsf{Dist}\Sigma_{\ell+1}^{\mathsf{P}} \subseteq \mathsf{AvgP}$. We will show a similar result for the class promise-$\Sigma_\ell^{\mathsf{BP}}$ defined below.

Definition 39 (promise-$\Sigma_\ell^{\mathsf{BP}}$ and promise-lin$\Sigma_\ell^{\mathsf{BP}}$)**.** A promise problem $\Pi = (\Pi_Y, \Pi_N) \in$ promise-$\Sigma_\ell^{\mathsf{BP}}$ for $\ell \geq 0$, if there is a $\Sigma_\ell \circ \mathsf{BP} \circ \mathsf{P}$ formula

$$\phi(x) = \exists\, y_1\, \forall\, y_2 \,\ldots\, Q\, y_\ell\; \mathsf{BP} r \;\; R(x, y_1, y_2, \ldots, y_\ell, r),$$

for a polytime predicate R and all string variables $y_1, \ldots, y_\ell, r$ of length polynomial in $|x|$, such that, for all $x \in \{0,1\}^n$,

$$\begin{aligned} x \in \Pi_Y \quad &\Longrightarrow \quad \exists\, y_1\, \forall\, y_2 \,\ldots\, Q\, y_\ell\; \mathsf{BP}_{(2/3)} r \;\; R(x, y_1, y_2, \ldots, y_\ell, r), \text{ and} \\ x \in \Pi_N \quad &\Longrightarrow \quad \forall\, y_1\, \exists\, y_2 \,\ldots\, \bar{Q}\, y_\ell\; \mathsf{BP}_{(2/3)} r \;\; \neg R(x, y_1, y_2, \ldots, y_\ell, r), \end{aligned}$$

where $\mathsf{BP}_{(\delta)} r \;\; R(-, r)$ means that $\mathbf{Pr}_r\,[R(-, r)] \geq \delta$. The constant $2/3$ in the definition above can be changed to $1 - 2^{-\mathsf{poly}(n)}$ by standard success amplification techniques (even if we start with any pair $\delta_Y > 1 - \delta_N + \varepsilon$ of thresholds, for some $\varepsilon \geq 1/\mathsf{poly}(n)$, where δ_Y is the probability $R(-,r)$ accepts, and δ_N the probability $\neg R(-,r)$ accepts).

The promise class $\mathsf{promise\text{-}lin}\Sigma_\ell^{\mathsf{BP}}$ is defined the same way as above, with an extra condition that, for some constant $a > 0$, $|y_i| \leq an$ for all $1 \leq i \leq \ell$ (but the randomness r may still be of any $\mathsf{poly}(n)$ length).

Theorem 40 (Witness Compression for PH)**.** *For any $\ell \geq 0$, suppose $\mathsf{Dist}\Sigma_{\ell+2}^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. Then*

$$\mathsf{promise\text{-}}\Sigma_\ell^{\mathsf{BP}} = \mathsf{promise\text{-}lin}\Sigma_\ell^{\mathsf{BP}}.$$

Proof. The proof is by induction on ℓ. The base case of $\ell = 0$ is trivially true.

For any $\ell > 0$, suppose the claim is true for $\ell - 1$. Consider an arbitrary promise problem $\Pi = (\Pi_Y, \Pi_N) \in \mathsf{promise\text{-}}\Sigma_\ell^{\mathsf{BP}}$. That is, for every $x \in \{0,1\}^n$,

$$\begin{aligned} x \in \Pi_Y &\implies \exists\, y\; \Pi_{\ell-1} z\; \mathsf{BP}_{(1-2^{-n})} r \;\; R(x,y,z,r), \\ x \in \Pi_N &\implies \forall\, y\; \Sigma_{\ell-1} z\; \mathsf{BP}_{(1-2^{-n})} r \;\; \neg R(x,y,z,r), \end{aligned}$$

for some polytime predicate R, where z is a sequence of $\ell - 1$ strings $z_1, \ldots, z_{\ell-1}$ under the corresponding $\ell - 1$ quantifiers of the $\Pi_{\ell-1}$ pattern, and $|y|, |r|, |z_i| \leq q(n)$, for some polynomial q, for all $1 \leq i \leq \ell - 1$.

As in the proof of the inclusion $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}}$ [Lau83], we replace the BP quantifier by the $\exists\forall$ pattern if ℓ is odd, or by the $\forall\exists$ pattern if ℓ is even. We get that there exists a polytime predicate R' such that, for every $x \in \{0,1\}^n$,

$$\begin{aligned} x \in \Pi_Y &\implies \exists\, y\; \Pi_{\ell-1} z\; Q_0 z' Q_1 z'' \;\; R'(x,y,z,z',z''), \\ x \in \Pi_N &\implies \forall\, y\; \Sigma_{\ell-1} z\; Q_1 z' Q_0 z'' \;\; \neg R'(x,y,z,z',z''), \end{aligned}$$

where z', z'' have the lengths polynomial in n and $Q_0 Q_1 = \exists\forall$ if ℓ is odd, and $Q_0 Q_1 = \forall\exists$ if ℓ is even.

Note that the above transformation shows that our promise problem $\Pi \in \mathsf{promise\text{-}}\Sigma_{\ell+1}^{\mathsf{P}}$. For any given $x \in \Pi_Y$, let y_x be the lexicographically first string y such that

$$\Pi_{\ell-1} z\; Q_0 z'\; Q_1 z''\; R'(x,y,z,z',z''). \tag{23}$$

Claim 41. *For x and y_x as above,*

$$\Pi_{\ell-1} z\; \mathsf{BP}_{(1/\mathsf{poly}(n))} r \;\; R(x,y_x,z,r).$$

Proof. Immediate from the properties of Lautemann's proof of $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}}$ [Lau83]. Recall that in that proof, the subformula $\mathsf{BP} r \;\; R(-,r)$ is replaced by

$$\exists u_1 \ldots u_k\; \forall r \;\; \vee_{i=1}^k \;\; R(-, r \oplus u_i),$$

for some $k \leq \mathsf{poly}(|r|)$, where $|u_i| = |r|$, for all $1 \leq i \leq k$, and $\oplus$ is the bit-wise XOR. If $\mathbf{Pr}_r\,[R(-,r)] < 1/k$, then such a collection of $u_1, \ldots, u_k$ cannot exist. Indeed, each "shifted" predicate $R(-, r \oplus u_i)$, for $1 \leq i \leq k$, accepts less than $1/k$ fraction of $r \in \{0,1\}^{|r|}$, and hence, by the union bound, the OR of any k such shifts accepts less than $k \cdot (1/k) = 1$ fraction of $r \in \{0,1\}^{|r|}$. $\square$

On the other hand, for every $x \in \Pi_N$ and every y, we have

$$\Sigma_{\ell-1} z\ \mathsf{BP}_{(1-2^{-n})} r\ \ \neg R(x,y,z,r).$$

Using Claim 41, by standard success amplification, for a random string r' of length $q'(n)$, for some polynomial q', and a polytime predicate R^{boost}, we get for every $x \in \{0,1\}^n$ and y_x as defined above,

$$x \in \Pi_Y \implies \Pi_{\ell-1} z\ \mathsf{BP}_{(1-2^{-n})} r'\ \ R^{boost}(x, y_x, z, r'), \tag{24}$$

$$x \in \Pi_N \implies \forall\, y\ \Sigma_{\ell-1} z\ \mathsf{BP}_{(1-2^{-n})} r'\ \ \neg R^{boost}(x, y, z, r'). \tag{25}$$

Claim 42. *For some universal constant $a > 0$ and some polynomial p', for every n-bit $x \in \Pi_Y$ with a witness y_x as defined above, we have*

$$\mathsf{rK}^{p'(n)}(y_x \mid x) \leq an.$$

Proof. Using Equation (23), we get by Item 2 of Lemma 34 (since $\mathsf{Dist}\Sigma_3^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$), that, for some polynomial p and all large enough t,

$$\mathsf{rK}^{p(t)}(y_x \mid x) \leq \mathsf{rK}^{t}(x) - \mathsf{rK}^{p(t)}(x) + 2\log p(t).$$

Since, for any t, $\mathsf{rK}^t(x) \leq O(n)$, the claim follows. □

Consider the formula

$$\phi(x) = \exists\, y' (|y'| \leq an)\ \Pi_{\ell-1} z\ \mathsf{BP} w\ \ R''(x, y', z, w), \tag{26}$$

where the predicate R'' is computed by the following polytime algorithm A:

> On input x, y', z, and a string $w = w'r'$, where w' is of length $p'(n)$, and r' is of length $q'(n)$, first run the universal machine $U(y', x, w')$ for $p'(n)$ steps, getting an output y. Then run $R^{boost}(x, y, z, r')$, accepting iff R^{boost} accepts.

We claim that Equation (26) is a correct $\Sigma_\ell^{\mathsf{BP}}$ formula for the promise problem Π. Indeed, for any $x \in \Pi_Y$, by Claim 42, there is a string y' that is decompressed to y_x with high probability over random strings w' (say, with probability at least $3/4$). By Equation (24), $R^{boost}(x, U(y', x, w'), z, r')$ accepts with probability at least $(3/4)(1 - 2^{-n}) > 2/3$ over strings $w = w'r'$. On the other hand, for any $x \in \Pi_N$, Equation (25) implies that

$$\forall\, y'\ \Sigma_{\ell-1} z\ \mathsf{BP}_{(1-2^{-n})} w\ \neg R''(x, y', z, w).$$

Next, using Equation (26), define a new promise problem $\Pi' = (\Pi'_Y, \Pi'_N) \in \mathsf{promise}\text{-}\Pi_{\ell-1}^{\mathsf{BP}}$:

$$\Pi'_Y = \left\{ (x, y') \mid |y'| \leq a|x|,\ \ \Pi_{\ell-1} z\ \mathsf{BP} w\ R''(x, y', z, w) \right\},$$
$$\Pi'_N = \left\{ (x, y') \mid |y'| \leq a|x|,\ \ \Sigma_{\ell-1} z\ \mathsf{BP} w\ \neg R''(x, y', z, w) \right\}.$$

By the Inductive Hypothesis, $\Pi' \in \mathsf{promise}\text{-}\mathsf{lin}\Pi_{\ell-1}^{\mathsf{BP}}$, with all quantified binary strings of length $O(|x| + |y'|) \leq O(n)$. It follows that $\Pi \in \mathsf{promise}\text{-}\mathsf{lin}\Sigma_\ell^{\mathsf{BP}}$. □

Theorem 43. *For any $\ell \geq 0$,*

$$\mathsf{Dist}\Sigma_{\ell+2}^{\mathsf{P}} \subseteq \mathsf{AvgBPP} \implies \mathsf{promise}\text{-}\Sigma_{\ell}^{\mathsf{P}} \subseteq \mathsf{promise}\text{-}\mathsf{BPTIME}[2^{O(n/\log n)}].$$

Proof. The proof is by induction on ℓ. The base case of $\ell = 0$ is trivially true. Consider any $\ell \geq 1$, and assume the claim for $\ell - 1$. Let $\Pi = (\Pi_Y, \Pi_N) \in \mathsf{promise}\text{-}\Sigma_{\ell}^{\mathsf{P}}$ be arbitrary. By Theorem 40, $\Pi \in \mathsf{promise}\text{-}\mathsf{lin}\Sigma_{\ell}^{\mathsf{BP}}$ via some formula

$$\phi(x) = \exists y \; \Pi_{\ell-1} z \; \mathsf{BP} r \;\; R(x, y, z, r),$$

with a polytime R, linearly-bounded y and $z = (z_1, \ldots, z_{\ell-1})$ (under the $\Pi_{\ell-1}$ quantifiers), and a poly-bounded r.

Arguing as in the proof of Theorem 40 (see the proof of Claim 42), but using Lemma 31 (for the case of pK) to bound the computational depth of an input x by $O(n/\log n)$ with a time bound $t \leq 2^{O(n/\log n)}$, we get that, for some universal constant $a > 0$, every $x \in \Pi_Y$ has a witness y_x with

$$\mathsf{pK}^{2^{an/\log n}}(y_x \mid x) \leq an/\log n. \tag{27}$$

Applying success amplification to R, we get a new polytime predicate R^{boost} with poly-bounded randomness r' such that we get for every $x \in \{0,1\}^n$ and y_x as defined above,

$$x \in \Pi_Y \implies \Pi_{\ell-1} z \; \mathsf{BP}_{(1-2^{-n})} r' \;\; R^{boost}(x, y_x, z, r'), \tag{28}$$

$$x \in \Pi_N \implies \forall y \; \Sigma_{\ell-1} z \; \mathsf{BP}_{(1-2^{-n})} r' \;\; \neg R^{boost}(x, y, z, r'). \tag{29}$$

Define a promise problem $\Pi' = (\Pi'_Y, \Pi'_N) \in \mathsf{promise}\text{-}\Pi_{\ell-1}^{\mathsf{BP}}$:

$$\Pi'_Y = \left\{(x, y) \mid \Pi_{\ell-1} z \; \mathsf{BP} r' \;\; R^{boost}(x, y, z, r')\right\},$$

$$\Pi'_N = \left\{(x, y) \mid \Sigma_{\ell-1} z \; \mathsf{BP} r' \;\; \neg R^{boost}(x, y, z, r')\right\}.$$

By the Inductive Hypothesis, $\Pi' \in \mathsf{promise}\text{-}\mathsf{BPTIME}[2^{O(n/\log n)}]$, where $|x| = n$, via some randomized algorithm A, with error probability at most 2^{-n}.

We now solve Π with the following randomized algorithm B:

> Given $x \in \{0,1\}^n$, choose a random string w of length $2^{an/\log n}$, and then enumerate over all machines $\mathcal{M} \in \{0,1\}^{an/\log n}$, and run $\mathcal{M}(w, x)$ for $2^{an/\log n}$ steps, getting a candidate witness y of length $O(n)$. Run $A(x, y)$, accepting iff A accepts.

The time complexity of the described algorithm B is clearly $2^{O(n/\log n)}$. For correctness, for every $x \in \Pi_Y$, with probability at least $3/4$ over w, our algorithm will consider a short description of a TM $\mathcal{M}$ such that $\mathcal{M}(w, x) = y_x$. Since $(x, y_x) \in \Pi'_Y$, we get that $A(x, y_x)$ will accept with probability at least $1 - 2^{-n}$ over its internal randomness. Hence, the algorithm B accepts x with probability at least $(3/4)(1 - 2^{-n}) > 2/3$. On the other hand, for every $x \in \Pi_N$, for every w and all $\mathcal{M}$, the string $y = \mathcal{M}(w, x)$ is such that $(x, y) \in \Pi'_N$, and so $A(x, y)$ accepts with probability at most 2^{-n}. It follows that the probability that $B(x)$ accepts in this case is

$$\mathop{\mathbf{E}}_{w,A}\left[\exists v \in \{0,1\}^{an/\log n} \; : \; A(x, U(v, x, w))\right] \leq 2^{an/\log n} \cdot \mathop{\mathbf{E}}_{w}\left[2^{-n}\right] \leq 2^{-n/2},$$

for all sufficiently large n. □

Corollary 44. *If* $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{PH} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$*. More precisely, for any* $\ell \geq 0$*,*

$$\mathsf{Dist}\Sigma^{\mathsf{P}}_{\ell+2} \subseteq \mathsf{AvgBPP} \implies \Sigma^{\mathsf{P}}_{\ell} \subseteq \mathsf{BPTIME}\left[2^{O(n/\log n)}\right].$$

Proof. Immediate from Theorem 43. □

4.5 Fine-grained case

Recall the quasilinear-time complexity classes $\mathsf{QL} = \mathsf{DTIME}[\widetilde{O}(n)]$, $\mathsf{NQL} = \mathsf{NTIME}[\widetilde{O}(n)]$, and the quasilinear-time analog $\mathsf{QLH} = \cup_{\ell\geq 0}\Sigma^{\mathsf{QL}}_{\ell}$ of the polynomial-time hierarchy PH; see, e.g., [NRS95] for more details. Define $\mathsf{BPQL} = \mathsf{BPTIME}[\widetilde{O}(n)]$, the quasilinear-time version of BPP. We show better probabilistic worst-case to average-case reductions under fine-grained average-case easiness assumptions.

Theorem 45. $\Sigma^{\mathsf{QL}}_2 \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL} \implies \mathsf{AMTIME}\left[2^{O(\sqrt{n\log n})}\right] = \mathsf{BPTIME}\left[2^{O(\sqrt{n\log n})}\right].$

It suffices to prove the inclusion $\mathsf{AMTIME}[2^{O(\sqrt{n\log n})}] \subseteq \mathsf{BPTIME}[2^{O(\sqrt{n\log n})}]$ (as the other inclusion is obvious). We will need the following analog of Lemma 31. The difference is that the "blow-up" of the time bound in pK^t here is *linear* in t, while in Lemma 31, this blow-up is *polynomial* in t.

Lemma 46 (Fine-Grained Computational Depth). *Let* $x \in \{0,1\}^n$ *and* $r \in \{0,1\}^*$ *be arbitrary strings,* f *any function, and* $\tau_n(t) := t \cdot \mathsf{poly}(n)$*. There exists a constant* $a > 0$ *and a time bound* t *such that* $f(n) \leq t \leq f(n)\cdot 2^{a\sqrt{n\log n}}$ *and*

$$\mathsf{pK}^t(x \mid r) - \mathsf{pK}^{\tau_n(t)}(x \mid r) \leq O\left(\sqrt{n\log n}\right).$$

Proof. Fix $x \in \{0,1\}^n$. Let $\tau_n(t) := t \cdot n^c$. Define $t_0 := f(n)$, and for $i \geq 1$, $t_i := \tau_n(t_{i-1})$. Consider the following telescoping sum:

$$\begin{aligned}\mathsf{pK}^{t_0}(x \mid r) - \mathsf{pK}^{t_I}(x \mid r) = &\left(\mathsf{pK}^{t_0}(x \mid r) - \mathsf{pK}^{t_1}(x \mid r)\right)\\ &+ \left(\mathsf{pK}^{t_1}(x \mid r) - \mathsf{pK}^{t_2}(x \mid r)\right) + \cdots + \left(\mathsf{pK}^{t_{I-1}}(x \mid r) - \mathsf{pK}^{t_I}(x \mid r)\right).\end{aligned}$$

Note that $\mathsf{pK}^{f(n)}(x \mid r) \leq n + d$, for some universal constant $d \geq 0$; hence, the above sum is at most $n + d$. By averaging, there is some index $1 \leq i_* \leq I$ such that

$$\mathsf{pK}^{t_{i_*-1}}(x \mid r) - \mathsf{pK}^{t_{i_*}(n)}(x \mid r) \leq \frac{n+d}{I},$$

which is at most $O\left(\sqrt{n\log n}\right)$ for $I := \sqrt{n/\log n}$. Finally, by induction, for every $i \geq 0$, $t_i \leq f(n)\cdot 2^{c\cdot i\cdot\log n}$. The bound $t_{i_*-1} \leq f(n)\cdot 2^{O(\sqrt{n\cdot\log n})}$ follows. □

We will also need the following fine-grained version of Lemma 26.

Lemma 47 (Fine-Grained Symmetry of Information for pK^t). *If* $\mathsf{NQL} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$*, then there exists a polynomial* p *such that for all sufficiently large* $n, m \in \mathbb{N}$*, all* $t \in \mathbb{N}$ *such that* $t \geq 2^{(n\cdot m)^{\varepsilon}}$ *for some* $\varepsilon > 0$*, and all time-constructible* $0 \leq \tau(n,m) \leq t$*, the following holds: for every* $x \in \{0,1\}^n$ *and every family* $\left\{y_r \in \{0,1\}^m \mid r \in \{0,1\}^{\tau(n,m)}\right\}$*, with probability at least* $9/10$ *over* $r \in \{0,1\}^{\tau(n,m)}$*,*

$$\mathsf{pK}^t(x, y_r \mid r) > \mathsf{pK}^{\widetilde{O}(t)}(x \mid r) + \mathsf{pK}^{\widetilde{O}(t)}(y_r \mid x, r) - \log p(t).$$

Proof Sketch. The proof is adapted from that of Lemma 26. Define

$$L := \left\{ (u, v, w, w', 1^t, 1^s, 1^n, 1^m) \;\middle|\; \begin{array}{l} \exists\, \mathcal{M} \in \{0,1\}^s, \quad \mathcal{M}(w, w') \text{ prints } uv \text{ within } 2t \text{ steps,} \\ \text{where } |w'| = \tau(n,m) \le t,\ |w| = 2t, \text{ and } s = |u| + |v| - 10 \end{array} \right\}.$$

A distribution D will be given parameters $\langle nk + k, mk' + k', 4t, s, n, m\rangle =: N$. It will be defined to randomly sample $u \sim \mathcal{U}_{nk+k}$, $v \sim \mathcal{U}_{mk'+k'}$, $w \sim \mathcal{U}_{2t}$, and $w' \sim \mathcal{U}_{\tau(n,m)}$, and then output $(u, v, w, w', 1^t, 1^s, 1^n, 1^m)$.

The key observation is that L can be solved in NQL, so using our fine-grained average-case easiness assumption, we obtain a randomized heuristic algorithm B that solves the distributional problem (L, D) in quasilinear time. In particular, since $t \ge 2^{(n\cdot m)^{\varepsilon}}$ for some $\varepsilon > 0$, $N \le \widetilde{O}(t)$. This ensures that B runs in time $\widetilde{O}(t)$ (see Proposition 16 and Definition 5).

The rest of the proof follows that of Lemma 26, except when B is used later on as a distinguisher, where we invoke the reconstruction lemma (Lemma 22), we get time bounds of $\widetilde{O}(t)$ instead of $\mathsf{poly}(t)$. $\square$

Using similar ideas, we can get an analog of Lemma 33 for a special case of $\ell = 1$. We will use the following definitions, similar to Definition 32, which respectively allow for larger time-bounds and impose a separate length restriction on witnesses.

Definition 48 ($\mathsf{BP}_\delta\circ\mathsf{NTIME}$ and $\mathsf{BP}_\delta\circ\mathsf{NTIMEGUESS}$)**.** A language L is in the class $\mathsf{BP}_\delta\circ\mathsf{NTIME}[\tau]$ for a time bound τ if there is a formula

$$\phi(x) = \mathsf{BP}r\ \exists y\ \ R(x, y, r)$$

for a predicate R running in time τ and variables y, r of length at most τ, such that, for all $x \in \{0,1\}^n$,

$$\begin{aligned} x \in L &\implies \Pr_r[\exists y\ \ R(x,y,r)] \ge \delta, \text{ and} \\ x \notin L &\implies \Pr_r[\forall y\ \ \neg R(x,y,r)] \ge \delta. \end{aligned}$$

The class $\mathsf{BP}_\delta \circ \mathsf{NTIMEGUESS}[\tau, \ell]$ is defined the same way as above, with an extra condition that $|y| \le \ell$ (but the randomness r may still be of length at most τ).

Lemma 49 (Fine-Grained Oracle Elimination)**.** *Suppose* $\mathsf{NQL} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$*. Let* $\tau(n) = 2^{O(\sqrt{n \log n})}$ *be some time-constructible function and* $c > 0$ *a constant. For an arbitrary* $L \in \mathsf{BP}_\delta \circ \mathsf{NTIMEGUESS}[\tau(n), n^c]$*, let* $x \in L_n$ *be sufficiently large, let* $r \in \{0,1\}^{\tau(n)}$ *be any random string under the* BP *quantifier such that there exists a witness for* $x \in L$*, and let* $y_{x,r} \in \{0,1\}^{n^c}$ *be the lexicographically first such* L*-witness for* x *and randomness* r*.*

There exist polynomials q *and* q_0 *such that, for all sufficiently large* $n \in \mathbb{N}$*, all* $x \in L_n$*, and all* $t \ge q_0(\tau(n))$*, we have with probability at least* $\delta - 1/10$ *over* $r \in \{0,1\}^{\tau(n)}$ *that an* L*-witness* $y_{x,r}$ *exists, and*

$$\mathsf{pK}^{\widetilde{O}(t)}(x, y_{x,r} \mid r) \le \mathsf{pK}^t(x \mid r) + \log q(t).$$

Proof Sketch. The proof is adapted from that of Lemma 33. Define

$$L' := \left\{ \left(\mathsf{DP}_k(x, y; z), w, w', 1^t, 1^k, 1^s, 1^n\right) \middle| \begin{array}{c} \exists \mathcal{M} \in \{0,1\}^s, \mathcal{M}^{\mathsf{SAT}}(w, w') \text{ prints } (x, y) \in \{0,1\}^{n+n^c} \\ \text{within } 2t \text{ steps,} \\ \text{where } |w'| = \tau(n) \le t,\ |w| = 2t, \text{ and } s = k - 10 \end{array} \right\}.$$

Since SAT is NQL-complete under many-one QL reduction, we have, for $x \in \{0,1\}^n$, $y \in \{0,1\}^{n^c}$, $w \in \{0,1\}^{2t}$, and $w' \in \{0,1\}^{\tau(n)}$ that $L' \in \Sigma_2^{\mathsf{QL}}$.

Again, using our fine-grained average-case easiness assumption, we get a randomized heuristic algorithm B that solves L' in time $\widetilde{O}(t)$. Also note that given x and good randomness $r \in \{0,1\}^{\tau(n)}$, we can compute $y_{x,r}$ via search-to-decision reduction using a SAT oracle, which takes time $\mathsf{poly}(\tau) =: q_0(\tau)$. Therefore, similar to Equation (19) in the original proof, we get

$$\mathsf{pK}^{2t,\mathsf{SAT}}(x, y_{x,r} \mid r) \le \mathsf{pK}^t(x \mid r) + d \log t \tag{30}$$

for some constant $d > 0$, provided $t \ge q_0(\tau)$. The rest of the proof is the same as that of Lemma 33, except when B is used later in the proof as a distinguisher, where we invoke the reconstruction lemma (Lemma 22), we get a time bound of $\widetilde{O}(t)$ instead of $\mathsf{poly}(t)$. □

We will also use the following simple generalization of (non-fine-grained) oracle elimination, Lemma 33, to higher time bounds.

Lemma 50. *Suppose* $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$. *For an arbitrary* $L \in \mathsf{NTIME}[\tau(n)]$, *for some* $\tau(n) = 2^{O(\sqrt{n \log n})}$, *let* $x \in L_n$ *be sufficiently large, and let* $y_x \in \{0,1\}^{\tau(n)}$ *be the lexicographically first* L*-witness for* x. *There exist polynomials* q *and* q_0 *such that, for all* $t \ge q_0(\tau(n))$, *we have*

$$\mathsf{pK}^{q(t)}(x, y_x) \le \mathsf{pK}^t(x) + \log q(t).$$

Proof Sketch. The proof follows that of the "Moreover" statement of Lemma 33, except the search-to-decision computation of y_x, where $|y_x| = \tau(n)$, now takes time at least $q_0(\tau(n))$ for some polynomial q_0, yielding

$$\mathsf{pK}^{2t,\mathsf{NP}}(x, y_{x,r} \mid r) \le \mathsf{pK}^t(x \mid r) + d \log n$$

for $t \ge q_0(\tau(n))$ in Equation (19). □

We are now ready to show Theorem 45.

Proof of Theorem 45. Assume $\Sigma_2^{\mathsf{QL}} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$. Let $\tau = 2^{a\sqrt{n \log n}}$ for some constant $a > 0$, and let $L \in \mathsf{AMTIME}[\tau]$ be arbitrary. That is, for every $x \in \{0,1\}^n$,

$$\begin{aligned} x \in L &\implies \mathsf{BP}_{(1-2^{-n})} r\ \exists y\ V(x, y, r), \\ x \notin L &\implies \mathsf{BP}_{(1-2^{-n})} r\ \forall y\ \neg V(x, y, r), \end{aligned}$$

for a polytime verifier $V(x, y, r)$, where $|r|, |y| = \tau$. Let $x \in L_n$ and good randomness $r \in \{0,1\}^\tau$ be given, with lexicographically first witness $y_{x,r} \in \{0,1\}^\tau$.

Note that $\Sigma_2^{\mathsf{QL}} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$ implies $\mathsf{Dist}\Sigma_2^{\mathsf{P}} \subseteq \mathsf{AvgBPP}$ by a simple padding argument. Combining Lemma 50 and Lemma 26 as in the proof of Lemma 34, for some constant $b > 0$, with probability at least $1 - 2^{-n} - 1/5$ over $r \in \{0,1\}^\tau$,

$$\mathsf{pK}^{\tau^b}(y_x \mid x, r) \leq \left(\mathsf{pK}^{\tau}(x \mid r) - \mathsf{pK}^{\tau^b}(x \mid r)\right) + O(\log \tau), \tag{31}$$

from which it follows that $\mathsf{pK}^{\tau^b}(y_x \mid x, r) \leq 2n$ by the trivial upper bound on $\mathsf{pK}^{\tau}(x|r)$. In other words, by definition of pK,

$$\Pr_{r' \sim \{0,1\}^{\tau^b}} \left[\exists \mathcal{M} \in \{0,1\}^{2n},\ \mathcal{M}(x, r, r') \text{ outputs } y_x \text{ within time } \tau^b\right] \geq 2/3. \tag{32}$$

Define a verifier $V'(x, z, r, r')$, which does the following:

> Simulate the machine described by z on input (x, r, r') for τ^b steps to produce some output y_z, and then return the output of $V(x, y_z, r)$.

Note that V' runs in time τ^c for some constant $c > 0$. For $x \in L$, there exists $z \in \{0,1\}^{2n}$ such that $V'(x, z, r, r')$ so long as Equation (31) and Equation (32) both hold, which occurs with probability at least $(2/3)(1 - 2^{-n} - 1/5) =: \delta > 1/2$ over r, r'. For $x \notin L$, $V'(x, z, r, r')$ rejects unless there is an L-witness $y_{x,r}$ under the original verifier V, which occurs with probability at most 2^{-n}. This leads to the following characterization of L:

$$\begin{aligned} x \in L &\implies \mathsf{BP}_{(\delta)} r, r' \ \exists z \ V'(x, z, r.r'), \\ x \notin L &\implies \mathsf{BP}_{(1-2^{-n})} r, r' \ \forall z \ \neg V'(x, z, r, r'), \end{aligned}$$

where $|r| = \tau$, $|r'| = \tau^b$, and $|z| = 2n$, which implies that $L \in \mathsf{BP}_\delta \circ \mathsf{NTIMEGUESS}[\tau^c, 2n]$.

Combining Lemma 47 and Lemma 49 as in the proof of Lemma 34, we have that with probability at least $\delta - 1/5 > 1/4$ over $(r, r') \in \{0,1\}^{\tau + \tau^b}$ that for all $t \geq p_0'(\tau^c)$, for some polynomials p' and p_0',

$$\mathsf{pK}^{t \cdot p'(n)}(z_{x,r,r'} \mid x, r, r') \leq \left(\mathsf{pK}^{t}(x \mid r, r') - \mathsf{pK}^{t \cdot p'(n)}(x \mid r, r')\right) + O(\log t) \tag{33}$$

where $z_{x,r,r'}$ is such that $V'(x, z_{x,r,r'}, r, r')$ accepts. Lemma 46 then implies the existence of some $t \in \mathbb{N}$ such that $p_0'(\tau^c) \leq t \leq 2^{O(\sqrt{n \log n})}$ and $\mathsf{pK}^{t}(x \mid r, r') - \mathsf{pK}^{t \cdot p'(n)}(x \mid r, r') \leq O(\sqrt{n \log n})$. So by Equation (33), for some constant $d > 0$,

$$\mathsf{pK}^{\tau^d}(z_{x,r} \mid x, r, r') \leq d\sqrt{n \log n}.$$

This suggests the following algorithm to decide L:

> Given $x \in \{0,1\}^n$, sample uniformly random strings $r \sim \{0,1\}^\tau$, $r' \in \{0,1\}^{\tau^b}$, and $w \in \{0,1\}^{\tau^d}$. Then exhaustively search over all $d\sqrt{n \log n}$-bit machines $\mathcal{M}$, running each $\mathcal{M}(x, r, r', w)$ for τ^d steps to produce a candidate witness z, accepting if $V'(x, z, r, r')$ accepts.

The runtime $\mathsf{poly}(\tau) = 2^{O(\sqrt{n \log n})}$ is clear. For correctness, for every $x \in L_n$, there is at least a $1/4$ fraction of good random strings r, r' such that Equation (33) holds, in which case a witness $z_{x,r,r'}$ will be found by our algorithm with probability at least $2/3$ over w (by definition of pK). So, our algorithm accepts $x \in L_n$ with probability at least $(1/4)(2/3) = 1/6$. On the other hand, for $x \notin L$, our algorithm will accept with probability at most 2^{-n}, as this upper-bounds the probability of r being good for the original verifier V. Thus, we have $L \in \mathsf{BPTIME}[2^{O\sqrt{n \log n}}]$ via standard success amplification. $\square$

A proof of the analogous statement for the class $\mathsf{UTIME}[2^{O(\sqrt{n \log n})}]$, Item 1 of Theorem 2,

$$\mathsf{NQL} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL} \implies \mathsf{UTIME}\left[2^{O(\sqrt{n \log n})}\right] \subseteq \mathsf{RTIME}\left[2^{O(\sqrt{n \log n})}\right],$$

is presented in Appendix B. Though its proof requires some additional ideas, it does not need to use an extremely efficient PRG as the one developed in [CHV22].

Finally, we note that one immediately obtains the main results of [CHV22], under *deterministic* average-case easiness assumptions, as corollaries of the theorems above via a simpler proof. For example, we will prove the following.

Theorem 51 ([CHV22])**.** $\Sigma_2^{\mathsf{QL}} \times \mathsf{QLSamp} \subseteq \mathsf{AvgQL} \implies \mathsf{AMTIME}\left[2^{O(\sqrt{n \log n})}\right] = \mathsf{DTIME}\left[2^{O(\sqrt{n \log n})}\right].$

Proof. If $\Sigma_2^{\mathsf{QL}} \times \mathsf{QLSamp} \subseteq \mathsf{AvgQL}$, then trivially $\Sigma_2^{\mathsf{QL}} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$, and by Theorem 45, we get that $\mathsf{AMTIME}[2^{O(\sqrt{n \log n})}] = \mathsf{BPTIME}[2^{O(\sqrt{n \log n})}]$. On the other hand, $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ (implied by the assumption of the theorem) yields $\mathsf{BPP} = \mathsf{P}$ [BFP05], which implies by padding that $\mathsf{BPTIME}[2^{O(\sqrt{n \log n})}] = \mathsf{DTIME}[2^{O(\sqrt{n \log n})}]$. $\square$

Remark 52. The main technical work in [CHV22] was to construct a particular efficient PRG to argue that the DPG reconstruction yields a *quasilinear*-time *deterministic* Kolmogorov complexity. Instead, by using the probabilistic time-bounded Kolmogorov complexity measure pK, we forgo the need for a PRG within the DPG reconstruction lemma. This yields a *randomized* algorithm for any given language in $\mathsf{NTIME}[2^{O(\sqrt{n \log n})}]$. Finally, at the very end, we derandomize this randomized algorithm, with polynomial overhead, using a standard hardness-based PRG [NW94; IW97], which is shown to exist by [BFP05] assuming $\mathsf{DistNP} \subseteq \mathsf{AvgP}$.

4.6 Time-computable heuristic schemes

One drawback of the worst-case to average-case reductions in Theorem 1 is that assuming average-case easiness for a class such as NP, we only get non-trivial worst-case algorithms for a smaller class, such as UP. A natural question then is whether we can get non-trivial worst-case algorithms for NP assuming only average-case easiness for NP. While this remains an interesting open problem, Hirahara [Hir21a] showed that such a worst-case to average-case reduction exists for NP if one considers a stronger notion of average-case easiness called P-*computable average-case polynomial time* (denoted as $\mathsf{Avg_P P}$), where, given an input x, the running time of the average-case algorithm on x can be efficiently estimated.

The proof of Hirahara's result built on ideas developed by [AF09], who showed that under a strong derandomization assumption, a language L is average-case easy *if and only if* it can be solved in time $2^{O\left(\mathsf{K}^{\mathsf{poly}(n)}(x) - \mathsf{K}(x) + \log n\right)}$ for every input $x \in \{0,1\}^n$. The proof of this result in [AF09] makes use of a fundamental theorem in Kolmogorov complexity called *language compression*, which states that for every (computable) language L, $\mathsf{K}(x) \lesssim \log |L \cap \{0,1\}^n|$ for all $x \in L \cap \{0,1\}^n$. The intuition is that for an average-case easy language, the set of "hard" instances that requires large running time must be small, and by the language compression theorem has low Kolmogorov complexity. This indicates that an instance with high (time-unbounded) Kolmogorov complexity has small running time.

Under the assumption $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, Hirahara proved a K^t version of the language compression theorem for languages in NP. By further assuming that one can efficiently estimate the running times of the average-case algorithms for NP, which is required to apply the ideas of [AF09] in the time-bounded case, he showed that NP can be solved in worst-case time $2^{O\left(\mathsf{K}^t(x) - \mathsf{K}^{\mathsf{poly}(t)}(x) + \log t\right)}$ for every input x and large enough t. As explained above, for every $x \in \{0,1\}^n$ there exists some $t \leq 2^{O(n/\log n)}$ such that $\mathsf{K}^t(x) - \mathsf{K}^{\mathsf{poly}(t)}(x) \leq O(n/\log n)$ so this yields non-trivial worst-case running times for NP. We remark that the reason why a language compression theorem only for NP is sufficient in the above argument is linked to the fact that K^t complexity can be checked in NP.

Here, we consider an analogy of $\mathsf{Avg_P P}$ in the randomized setting, called $\mathsf{Avg_{BPP}BPP}$, and we show that $\mathsf{DistNP} \subseteq \mathsf{Avg_{BPP}BPP}$ implies $\mathsf{NP} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$. While it is not surprising that our proof will require to show a pK^t version of the language compression theorem (under probabilistic average-case easiness of NP), there are also subtleties in terms of which language class that we need to compress. First of all, using ideas from previous sections, we can show that for every $L \in \mathsf{NP}$, we have $\mathsf{pK}^{\mathsf{poly}(n)}(x) \lesssim \log |L \cap \{0,1\}^n|$ for all $x \in L \cap \{0,1\}^n$. However, it turns out that such a language compression theorem for NP is not sufficient, and we actually need language compression for $\mathsf{promiseAM}$ (Section 3.6), again, due to the fact that pK^t is a more complicated notion and can only be estimated in $\mathsf{promiseAM}$.

Definition 53 ($\mathsf{Avg_{BPP}BPP}$)**.** For a language L and a family $D = \{D_n\}_{n \in \mathbb{N}}$ of distributions, we say that $(L, D) \in \mathsf{Avg_{BPP}BPP}$ if there exist a randomized algorithm A, a function $t\colon \{0,1\}^* \times \{1\}^* \to \mathbb{N}$ and constants $\varepsilon, b > 0$ such that for every $n \in \mathbb{N}$,

1. $\mathbf{Pr}_A\left[A(x, 1^n) = L(x)\right] \geq 2/3$, for every $x \in \mathsf{Supp}(D_n)$,
2. $A(x, 1^n)$ halts within time $t(x, 1^n)$ (on each of its computational path), for every $x \in \mathsf{Supp}(D_n)$,
3. $\mathbf{E}_{x \sim D_n}\left[\frac{t(x,1^n)^\varepsilon}{n}\right] \leq b$, and
4. there is a polynomial-time randomized algorithm such that given x, 1^n and θ, decides whether $t(x, 1^n) \leq \theta$.

Theorem 54. *If* $\mathsf{DistNP} \subseteq \mathsf{Avg_{BPP}BPP}$, *then* $\mathsf{AM} \subseteq \mathsf{BPTIME}\left[2^{O(n/\log n)}\right]$.

For the proof of this result, we will also need the following weak symmetry of information, which follows as a simple corollary from Lemma 47 and Lemma 20.

Lemma 55 (Weak Symmetry of Information for pK^t)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, *then there exist polynomials* p_0 *and* p_w *such that for every sufficiently large* $x \in \{0,1\}^n$, $m \in \mathbb{N}$ *and* $p_0(n, m) \leq t \leq 2^{n+m}$,

$$\Pr_{u \sim \{0,1\}^m}\left[\mathsf{pK}^t(x, u) > \mathsf{pK}^{t \cdot p_w(nm)}(x) + m - \log p_w(t)\right] \geq 0.99.$$

We are now ready to prove Theorem 54.

Proof of Theorem 54. **A "universal" distribution.** We first define a distribution family $D = \{D_{\langle m,t_0\rangle}\}$, each of which does the following. On input $1^{\langle m,t_0\rangle}$,

1. Sample $s \in [2m]$,
2. Sample $w \in \{0,1\}^{t_0}$,
3. Sample $\mathcal{M} \in \{0,1\}^s$,
4. Run $\mathcal{M}(w)$ for t_0 steps, if it outputs a string y of length m, then output x; otherwise output 1^m.

Note that D is polynomial-time samplable. Moreover, it *dominates* the "universal distribution" for pK^{t_0}, which assigns each y the probability mass $2^{-\mathsf{pK}^{t_0}(y)}$. In particular, it is easy to see that for every $y \in \{0,1\}^m$,

$$D_{\langle m,t_0\rangle}(y) \geq \frac{2^{-\mathsf{pK}^{t_0}(y)}}{O(m)}. \tag{34}$$

Worst-case running times for NP. Let $L \in \mathsf{AM}$ be the language that we want to solve in the worst case. Let $L' \in \mathsf{NP}$ and $c > 0$ be a constant such that for every input $x \in \{0,1\}^n$ to L,

$$\Pr_{r\in\{0,1\}^{n^c}}\left[L'(x,r) = L(x)\right] \geq 1 - 1/n.$$

Therefore, to decide $L(x)$ (probabilisticaly), it suffices to decide $L'(x,r)$ for a typical r. For the rest of the proof, we will use x to denote an input to L and $y := (x,r)$ to L'.

By assumption, $(L', D) \in \mathsf{Avg_{BPP}BPP}$. Let A be an algorithm that solves L' on average with respect to D. Let $t_A\big(y, 1^{\langle m,t_0\rangle}\big)$ denote the running time of A on input $y \in \mathsf{Supp}(D_{\langle m,t_0\rangle})$. We will also write $t_A(y)$ instead of $t_A\big(y, 1^{\langle m,t_0\rangle}\big)$ when the input size to the distribution is clear in the context.

Let B be a $\mathsf{promiseBPP}$ algorithm that solves the promise problem from Lemma 27 and let τ be the polynomial there. Now, consider the following ensemble of promise problems:

$$\Pi_{\mathsf{YES}} := \left\{ \left(y, 1^{\langle m,i,s,t\rangle}\right) \;\middle|\; |y| = m,\;\; t_A\left(y, 1^{\langle m,\tau(t)\rangle}\right) \in \left[2^i, 2^{i+1}\right] \text{ and } B\big(y, 1^s, 1^t\big) = 1 \text{ w.p.} \geq 2/3 \right\},$$

$$\Pi_{\mathsf{NO}} := \left\{ \left(y, 1^{\langle m,i,s,t\rangle}\right) \;\middle|\; |y| = m,\;\; t_A\left(y, 1^{\langle m,\tau(t)\rangle}\right) \notin \left[2^i, 2^{i+1}\right] \text{ or } B\big(y, 1^s, 1^t\big) = 0 \text{ w.p.} \geq 2/3 \right\}.$$

Note that the above problem is in $\mathsf{PromiseBPP}$, since B is a $\mathsf{promiseBPP}$ algorithm and t_A is probabilistically polynomial-time checkable. Let

$$H_{\langle m,i,s,t\rangle} := \{0,1\}^m - \Pi_{\mathsf{NO},\langle m,i,s,t\rangle}.$$

Note that by property of B (recall Lemma 27), for every $y \in H_{\langle m,i,s,t\rangle}$, we have

$$\mathsf{pK}^{\tau(t)}(y) < s + \log \tau(t). \tag{35}$$

Now consider any $m, i, s, t \in \mathbb{N}$. Since A runs in time polynomial on average with respect to D, there are constants b and ε such that

$$\begin{aligned}
b &\geq \sum_{y \in H_{\langle m,i,s,t\rangle}} \frac{t_A(y)^{\varepsilon}}{\langle m, \tau(t)\rangle} \cdot D_{\langle m,\tau(t)\rangle}(y) \\
&\geq \sum_{y \in H_{\langle m,i,s,t\rangle}} \frac{t_A(y)^{\varepsilon}}{\langle m, \tau(t)\rangle} \cdot \frac{2^{-\mathsf{pK}^{\tau(t)}(y)}}{O(m)} && \text{(Equation (34))} \\
&\geq |H_{\langle m,i,s,t\rangle}| \cdot \frac{2^{\varepsilon \cdot i}}{\tau(t)^3} \cdot 2^{-s-\log \tau(t)} && \text{(Equation (35))} \\
&= 2^{\varepsilon \cdot i + \log(|H_{\langle m,i,s,t\rangle}|) - s - 4\log \tau(t)}.
\end{aligned}$$

By rearranging, we get that for every $x \in H_{\langle m,i,s,t\rangle}$,

$$\varepsilon \cdot i \leq s - \log |H_{\langle m,i,s,t\rangle}| + O(\log t). \tag{36}$$

Also, applying language compression (Theorem 30) to $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$, we have that for every $y \in \Pi_{\mathsf{YES},\langle n,i,s,t\rangle} \subseteq H_{\langle m,i,s,t\rangle}$,

$$\mathsf{pK}^{p_{\mathsf{LC}}(\langle m,i,s,t\rangle)}(y) \leq \log |H_{\langle m,i,s,t\rangle}| + \log p_{\mathsf{LC}}(\langle m, i, s, t\rangle), \tag{37}$$

where p_{LC} is some polynomial. Combining Equations (36) and (37), and using the fact that $t_A\big(y, 1^{\langle m,\tau(t)\rangle}\big) \leq 2^{i+1}$ for every $y \in \Pi_{\mathsf{YES},\langle m,i,s,t\rangle}$, we have for every such y,

$$t_A\Big(y, 1^{\langle m,\tau(t)\rangle}\Big) \leq 2^{\varepsilon^{-1}\cdot\left(s - \mathsf{pK}^{p_1(\langle m,i,s,t\rangle)}(y) + \log p_1(\langle m,i,s,t\rangle)\right)},$$

where p_1 is some polynomial.

Worst-case running times for AM. Now fix any input $x \in \{0,1\}^n$ to L, and let $m := |(x, r)|$ for $r \in \{0,1\}^{n^c}$. Consider any $r \in \{0,1\}^{n^c}$. Let $s := \mathsf{pK}^t(x, r) \leq m^2$ and let i be an integer such that

$$2^i \leq t_A\Big((x,r), 1^{\langle m,\tau(t)\rangle}\Big) \leq 2^{i+1}.$$

By definition, $(x, r) \in \Pi_{\mathsf{YES},\langle m,i,s,t\rangle}$. Note that $i \leq m^2$, since A runs in polynomial time on average and every string y of length m in the support of the distribution has probability mass at least $2^{-O(m)}$. Consequently, from the above, we have for every $t \geq m^2$,

$$t_A\Big((x,r), 1^{\langle m,\tau(t)\rangle}\Big) \leq 2^{O\left(\mathsf{pK}^t(x,r) - \mathsf{pK}^{p_2(t)}(x,r) + \log p_2(t)\right)}, \tag{38}$$

where p_2 is some polynomial. Now observe that by weak symmetry of information (Lemma 55), as long as t is greater than $q(m)$ for some polynomial q, with probability at least 0.99 over r (which we consider *good*), we have

$$\begin{aligned}
&\mathsf{pK}^t(x,r) - \mathsf{pK}^{p_2(t)}(x,r) \\
&\leq \Big(\mathsf{pK}^{t/2}(x) + |r| + O(\log n)\Big) - \Big(\mathsf{pK}^{p_w(p_2(t))}(x) + |r| + \log p_w(p_2(t))\Big) \\
&\leq \mathsf{pK}^{t/2}(x) - \mathsf{pK}^{p_3(t/2)}(x) + \log p_3(t/2),
\end{aligned} \tag{39}$$

where p_3 is some polynomial. Then by Equation (38), Equation (39), and Lemma 31, there exist a constant $d > 0$ and $t^* \le 2^{dn/\log n}$ such that for every good r,

$$t_A\Big((x,r), 1^{\langle m,\tau(t^*)\rangle}\Big) \le 2^{dn/\log n}.$$

This suggests the following randomized algorithm for solving L.

> Given $x \in \{0,1\}^n$, randomly pick $r \sim \{0,1\}^{n^c}$ and let $m := |(x,r)|$. For $t = q(m), \ldots, 2^{dn/\log n}$, check if $t_A\big((x,r), 1^{\langle m,\tau(t)\rangle}\big) \le 2^{dn/\log n}$. If so, run $A\big((x,r), 1^{\langle n,\tau(t^*)\rangle}\big)$ for at most $2^{dn/\log n}$ steps and output whatever it outputs.

It is easy to verify that the above probabilistic algorithm runs in time $2^{O(n/\log n)}$ and decides L. $\square$

5 Learning in Randomized Heuristica

The main result of this section is the following.

Theorem 56. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, *then for any time constructible functions* $s, T, a\colon \mathbb{N} \to \mathbb{N}$, *and* $\varepsilon \in [0,1]$, $\mathsf{SIZE}[s(n)]$ *is agnostic learnable on* $\mathsf{Samp}[T(n)]/a(n)$ *in time* $\mathsf{poly}\big(n, \varepsilon^{-1}, s(n), T(n), a(n)\big)$ *with sample complexity*

$$\left(\frac{(n + s(n) + a(n) + \log T(n))^3}{\varepsilon^8}\right)^{1+o(1)}.$$

5.1 Ingredients

In this section, we will use D to denote a family of distributions (one for each n) and $\mathcal{D}$ to denote a class of family of distributions (e.g., those that are efficiently samplable). For a distribution D and $m \in \mathbb{N}$, let D^m denote the distribution $x_1 \circ x_2 \circ \cdots \circ x_m$ where $x_1, x_2, \ldots, x_m \sim D$.

5.1.1 Learning from RRHS Refutation

Definition 57 (Correlative RRHS-Refutation [KL18, Definition 3])**.** Let $\mathcal{C}$ be a concept class, and let $\mathcal{D}$ be a class of example distributions. A randomized algorithm A is a correlatively random-right-hand-side-refuter (correlative RRHS-refuter) for $\mathcal{C}$ on $\mathcal{D}$ with sample complexity m if A satisfies the following. A takes as input $n \in \mathbb{N}$, $\varepsilon \in (0,1)$ and a set $S = \big(\langle x^{(1)}, b^{(1)}\rangle, \ldots, \langle x^{(m)}, b^{(m)}\rangle\big)$ of samples, where $x^{(i)} \in \{0,1\}^n$ and $b^{(i)} \in \{0,1\}$ for every $i \in [m]$, and

- **Soundness:** if the samples S are i.i.d. from a distribution D' on $\{0,1\}^n \times \{0,1\}$ such that the marginal on $\{0,1\}^n$ equals D_n for some $D_n \in \mathcal{D}_n$ and there exists $f \in \mathcal{C}_n$ with

$$\Pr_{\langle x^{(i)}, b^{(i)}\rangle \sim D'}\Big[b^{(i)} = f(x^{(i)})\Big] \ge \frac{1}{2} + \frac{\varepsilon}{2},$$

  then

$$\Pr_{S,A}[A(n,\varepsilon,S) = \mathsf{correlative}\,] \ge 2/3.$$

- **Completeness:** if the samples S are selected i.i.d. so that $x^{(1)}, \dots, x^{(m)} \sim D_n$ for some $D_n \in \mathcal{D}_n$ and $b^{(1)}, \dots, b^{(m)} \sim \mathcal{U}$, then

$$\Pr_{S,A}\left[A(n, \varepsilon, S) = \textsf{random}\ \right] \geq 2/3.$$

Theorem 58 (Agnostic Learning from RRHS-Refutation [KL18]; see also [HN21, Theorem 5])**.** *Let $\mathcal{C}$ be a concept class, and let $\mathcal{D}$ be a class of example distributions. If there exists a correlative RRHS-refuter for $\mathcal{C}$ on $\mathcal{D}$ with sample complexity $m(n,\varepsilon)$ and running time $T(n,\varepsilon)$, then $\mathcal{C}$ is agnostic learnable with*

$$\text{sample complexity} = O\left(\frac{m(n,\varepsilon/2)^3}{\varepsilon^2}\right) \quad \textit{and} \quad \text{running time} = O\left(T(n,\varepsilon/2)\cdot\frac{m(n,\varepsilon/2)^2}{\varepsilon^2}\right).$$

5.1.2 Probabilistic sampling depth

Definition 59 (Probabilistic Sampling Depth)**.** Let $t, t' \in \mathbb{N}$, where $t' > t$. For a family $D = \{D_n\}_{n\geq 1}$ of distributions, we introduce the (t,t')*-probabilistic-sampling-depth functions* $\mathsf{psd}_D^{t,t'} := \left\{\mathsf{psd}_{D,n}^{t,t'}\right\}_{n\geq 1}$, where

$$\mathsf{psd}_{D,n}^{t,t'}(m) = \mathop{\mathbf{E}}_{X\sim D_n^m}\left[\mathsf{pK}^t(X) - \mathsf{pK}^{t'}(X)\right].$$

For a collection $\mathcal{D}$ of families of distributions, we define $\mathsf{psd}_{\mathcal{D}}^{t,t'} := \left\{\mathsf{psd}_{\mathcal{D},n}^{t,t'}\right\}_{n\geq 1}$, where

$$\mathsf{psd}_{\mathcal{D},n}^{t,t'}(m) = \max_{D\in\mathcal{D}}\ \mathsf{psd}_{D,n}^{t,t'}(m).$$

Lemma 60 (Small Probabilistic Sampling Depth for Samplable Distributions)**.** *There exists a polynomial $p_1 \colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ such that for any $T, a\colon \mathbb{N}\to\mathbb{N}$ and $n, m \in \mathbb{N}$, the following holds. For every $t \geq p_1(T(n), m)$, and every $t' > t$,*

$$\mathsf{psd}_{\mathsf{Samp}[T(n)]/a(n),n}^{t,t'}(m) \leq O\big(\log m + \log T(n) + a(n) + \log t'\big).$$

Proof. The proof is essentially the same as that of [HN21, Lemma 6], but uses the unconditional coding theorem for *probabilistic* Kolmogorov complexity (Lemma 25).

Fix any $D_n \in \mathsf{Samp}[T(n)]/a(n)$. Let p_1 be the polynomial p in Lemma 25. We have for every $t \geq p_1(T(n), m)$,

$$\begin{aligned}
\mathop{\mathbf{E}}_{x\sim D_n^m}\left[\mathsf{pK}^t(x)\right] &\leq \mathop{\mathbf{E}}_{x\sim D_n^m}\left[\mathsf{pK}^{p_1(T(n),m)}(x)\right] \\
&\leq \mathop{\mathbf{E}}_{x\sim D_n^m}\left[\log(1/D_n^m(x))\right] + O(\log(m) + \log T(n) + a(n)) && \text{(Lemma 25)} \\
&= H(D_n^m) + O(\log(m) + \log T(n) + a(n)) \\
&\leq \mathop{\mathbf{E}}_{x\sim D_n^m}\left[\mathsf{K}(x)\right] + O(\log(m) + \log T(n)) + a(n) \\
&\leq \mathop{\mathbf{E}}_{x\sim D_n^m}\left[\mathsf{pK}^{t'}(x)\right] + O\big(\log(m) + \log T(n) + a(n) + \log t'\big), && \text{(Lemma 18)}
\end{aligned}$$

where the second last inequality uses the fact that for any distribution D, its Shannon entropy $H(D) \leq \mathbf{E}_{y \sim D}[\mathsf{K}(y)]$ (see [LV19b, Theorem 8.1.1] and [HN21, Lemma 5]). Rearranging the above, we get

$$\mathop{\mathbf{E}}_{x \sim D_n^m}\left[\mathsf{pK}^t(x) - \mathsf{pK}^{t'}(x)\right] \leq O\big(\log(m) + \log T(n) + a(n) + \log t'\big)\,,$$

as desired. □

5.2 RRHS refuters from probabilistic average-case easiness

We first prove the following technical theorem.

Theorem 61. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, then for any time constructible functions* $s, T, a\colon \mathbb{N} \to \mathbb{N}$ *and for any constant* $\zeta > 0$*, there is a correlative RRHS-refuter for* $\mathsf{SIZE}[s(n)]$ *under* $\mathsf{Samp}[T(n)]/a(n)$ *with sample complexity*

$$m := \left(\frac{n + s(n) + a(n) + \log T(n)}{\varepsilon^2}\right)^{1+\zeta}$$

and running time $\mathsf{poly}(n, m, T(n)) = \mathsf{poly}(n, 1/\varepsilon, T(n), a(n), s(n))$*.*

Proof. The proof closely follows that of [HN21, Theorem 8].

The (correlative) RRHS-refuter R. Let τ be the polynomial from Lemma 28. Let $\ell_s(n) \leq O(s(n)\log s(n))$ denote the number of bits needed to encode a function $f \in \mathsf{SIZE}[s(n)]$. On input $n \in \mathbb{N}$, $\varepsilon > 0$, and a set $S = \left(\langle x^{(1)}, b^{(1)}\rangle, \ldots, \langle x^{(m)}, b^{(m)}\rangle\right)$ of samples, R operates as follows.

1. Compute $t := \max\{p_0\big(n \cdot m^2\big), p_1(T(n), m)\}$, where p_0 is the polynomial from Lemma 26 and p_1 is the polynomial from Lemma 60. Also compute $t' := t'(t)$, where t' is some polynomial specified later (in Claim 62).

2. Compute

$$\beta := \mathsf{Approx}_\tau\text{-}\mathsf{pK}\left(X, 1^t\right), \text{ and}$$
$$\beta' := \mathsf{Approx}_\tau\text{-}\mathsf{pK}\left(X \circ b, 1^{t'}\right),$$

   where $X := x^{(1)} \circ \cdots \circ x^{(m)}$, $b := b^{(1)} \circ \cdots \circ b^{(m)}$, and $\mathsf{Approx}_\tau\text{-}\mathsf{pK}$ is the randomized algorithm from Lemma 28.

3. Output "correlative" if $\beta' - \beta \leq \theta$, where $\theta := m(1 - \varepsilon^2/8) + \ell_s(n) + n + \log \tau(t)$, and output "random" otherwise.

It is easy to verify that the running time of R is $\mathsf{poly}(n, m, T(n))$. Next, we argue its correctness.

Soundness. Suppose we are in the "correlative" case. That is, the labelled examples in S are sampled i.i.d from some distribution D' on $\{0,1\}^n \times \{0,1\}$, whose marginal on $\{0,1\}^n$ is given by some $D \in \mathsf{Samp}[T(n)]/a(n)$, and there exists $f \in \mathsf{SIZE}[s(n)]$ such that

$$\mathop{\mathbf{Pr}}_{\langle x^{(i)}, b^{(i)}\rangle \sim D'}\left[b^{(i)} = f(x^{(i)})\right] \geq \frac{1}{2} + \frac{\varepsilon}{2}.$$

In this case, by a standard concentration bound, the probability over $S \sim (D')^m$ that

$$\left|\{i \in [m] \mid b_i = f(x^{(i)})\}\right| < (1/2 + \varepsilon/4) \cdot m$$

is at most $\exp(-2m(\varepsilon/4)^2) \le o(1)$, where the last inequality relies on our choice of m. Now observe the following.

Claim 62. There exists a polynomial t' such that for any $b \in \{0,1\}^m$ satisfying

$$\left|\{i \in [m] \mid b_i = f(x^{(i)})\}\right| \ge (1/2 + \varepsilon/4) \cdot m,$$

we have

$$\mathsf{pK}^{t'(t)}(X \circ b) \le \mathsf{pK}^{\tau(t)}(X) + \ell_s(n) + \left(1 - \varepsilon^2/8\right) \cdot m.$$

Proof of Claim 62. Note that given X, we can compute $f(x^{(1)}), \ldots, f(x^{(m)})$ in time $\mathsf{poly}(m \cdot \ell_s(n))$ using the encoding of f, which is of $\ell_s(n)$ bits. Note that b and $f(x^{(1)}), \ldots, f(x^{(m)})$ disagree on at most $(1/2 - \varepsilon/4) \cdot m$ coordinates. Then to recover b, we can define a string $e \in \{0,1\}^m$ such that $e_i = 1$ iff $f(x^{(i)}) \ne b_i$. Note that e has hamming weight at most $(1/2 - \varepsilon/4) \cdot m$. Using the inequality

$$\sum_{i=0}^{k} \binom{m}{i} \;\le\; 2^{H_2(k/m) \cdot m},$$

where H_2 is the binary entropy function, e can be encoded with $H_2(1/2 + \varepsilon/4) \cdot m$ bits, by lexicographic indexing among binary strings of bounded hamming weight. Using the Taylor series of H_2 in the neighborhood of $1/2$, for $\delta := \varepsilon/4$, we have

$$H_2(1/2 + \delta) = 1 - \frac{1}{2\ln 2} \sum_{i=1}^{\infty} \frac{(2\delta)^{2i}}{i(2i-1)} \le 1 - \frac{2}{\ln 2}\delta^2 \le 1 - 2\delta^2.$$

Moreover, using such an encoding, it is not hard to see that e can be reconstructed in time $\mathsf{poly}(n, m)$. Therefore, given X, we can compute b in time $\mathsf{poly}(n, m)$ using strings of length $\ell_s(n)$ (which encodes f) and $(1 - \varepsilon^2/8) \cdot m$ (which encodes e). In other words, if X has probabilistic $\tau(t)$-time-bounded Kolmogorov complexity k, then

$$\mathsf{pK}^{t'(t)}(X \circ b) \le k + \ell_s(n) + \left(1 - \varepsilon^2/8\right) \cdot m,$$

for some $t'(t) := \mathsf{poly}(\tau(t))$. This completes the proof of the claim. $\square$

Now, suppose in Step 2 of R, β and β' output by the algorithm Approx_τ-pK are good approximations as given in Lemma 28. Note that this happen with high probability. Then by a union bound, with probability at least 2/3, over the samples $S \sim (D')^m$ and the internal randomness of R, we have

$$\begin{aligned}
\beta' &\le \mathsf{pK}^{t'}(X \circ b) && (\beta' \text{ is a good approximation}) \\
&\le \mathsf{pK}^{\tau(t)}(X) + \ell_s(n) + \left(1 - \varepsilon^2/8\right) \cdot m && (\text{Claim } 62) \\
&\le \beta + \log \tau(t) + \ell_s(n) + \left(1 - \varepsilon^2/8\right) \cdot m, && (\beta \text{ is a good approximation})
\end{aligned}$$

which implies

$$\beta' - \beta \leq \bigl(1 - \varepsilon^2/8\bigr) \cdot m + \ell_s(n) + n + \log \tau(t) = \theta,$$

and R will output "correlative".

Completeness. Suppose we are in the "random" case. That is, b is sampled from $\mathcal{U}_m$. Assuming $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, by combining symmetry of information (Lemma 26) and probabilistic incompressibility (Lemma 20), for any $X \in \{0,1\}^{nm}$ and some polynomial p_w, we have

$$\Pr_{b \sim \mathcal{U}_m}\Bigl[\mathsf{pK}^{\tau(t')}(X \circ b) \geq \mathsf{pK}^{p_w(\tau(t'))}(X) + (m-10) - \log p_w\bigl(\tau(t')\bigr)\Bigr] \geq 1 - 1/8. \tag{40}$$

Let us define

$$p_\tau(t) := p_w(\tau(t'(t))).$$

By Markov, we have

$$\begin{aligned}&\Pr_{X \sim D^m}\Bigl[\mathsf{pK}^t(X) - \mathsf{pK}^{p_\tau(t)}(X) > 8 \cdot \mathsf{psd}^{t,p_\tau(t)}_{D,n}(m)\Bigr] \\ &\leq \frac{\mathbf{E}_{X \sim D^m}\Bigl[\mathsf{pK}^t(X) - \mathsf{pK}^{p_\tau(t)}(X)\Bigr]}{8 \cdot \mathsf{psd}^{t,p_\tau(t)}_{D,n}(m)} = \frac{1}{8}.\end{aligned} \tag{41}$$

Again, let us assume that in Step 2 of R, β and β' output by the algorithm Approx_τ-pK are good approximations. Then with with probability at least $1 - (1/8 + 1/8 + o(1)) \geq 2/3$ over $b \sim \mathcal{U}_m$, $X \sim D^m$ and the internal randomness of R, we have

$$\begin{aligned}\beta' &\geq \mathsf{pK}^{\tau(t')}(X \circ b) - \log \tau(t') && (\beta' \text{ is a good approximation})\\ &\geq \mathsf{pK}^{p_w(\tau(t'))}(X) + m - \log\left(p_w\left(\tau(t')\right) \cdot \tau(t')\right) && (\text{Equation (40)})\\ &= \mathsf{pK}^t(X) - \Bigl(\mathsf{pK}^t(X) - \mathsf{pK}^{p_\tau(t)}(X)\Bigr) + m - 10 - \log\bigl(p_\tau(t) \cdot \tau(t')\bigr)\\ &\geq \beta - 8 \cdot \mathsf{psd}^{t,p_\tau(t)}_{D,n}(m) + m - 10 - \log\bigl(p_\tau(t) \cdot \tau(t')\bigr), && (\text{Equation (41)})\end{aligned}$$

which implies

$$\beta' - \beta \geq m - 8 \cdot \mathsf{psd}^{t,p_\tau(t)}_{D,n}(m) - \log\bigl(p_\tau(t) \cdot \tau(t')\bigr) - 10.$$

Now we want the above to be greater than θ. We have

$$\begin{aligned}&(\beta' - \beta) - \theta\\ &\geq \Bigl(m - 8 \cdot \mathsf{psd}^{t,p_\tau(t)}_{D,n}(m) - \log\bigl(p_\tau(t) \cdot \tau(t')\bigr) - 10\Bigr)\\ &\qquad - \bigl(m + \ell_s(n) + n + \log \tau(t) - m\varepsilon^2/8\bigr)\\ &= m\varepsilon^2/8 - \Bigl(n + \ell_s(n) + 8 \cdot \mathsf{psd}^{t,p_\tau(t)}_{D,n}(m) + \log\bigl(p_\tau(t) \cdot \tau(t') \cdot \tau(t)\bigr) + 10\Bigr)\\ &\geq m\varepsilon^2/8 - \left[n + \ell_s(n) + c \cdot (\log m + \log T(n) + a(n)) + \log p_2(n, m, T(n)) + 10\right], \qquad (\text{Lemma 60})\end{aligned}$$

where $c > 0$ is some constant and p_2 is some polynomial (that depends on τ, p_0, p_1 and p_w). From here, it can be easily seen that the above is greater than 0 by our choice of m, provided that n is sufficiently large. Hence R will output "random". This completes the proof of the theorem. $\square$

5.3 Concluding the proof

We are now ready to prove Theorem 56.

Theorem 63 (Reminder of Theorem 56)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$*, then for any time constructible functions* $s, T, a \colon \mathbb{N} \to \mathbb{N}$*, and* $\varepsilon \in [0,1]$*,* $\mathsf{SIZE}[s(n)]$ *is agnostic learnable on* $\mathsf{Samp}[T(n)]/a(n)$ *in time* $\mathsf{poly}\big(n, \varepsilon^{-1}, s(n), T(n), a(n)\big)$ *with sample complexity*

$$\left(\frac{(n + s(n) + a(n) + \log T(n))^3}{\varepsilon^8}\right)^{1+\zeta},$$

where $\zeta > 0$ *is any arbitrary small constant.*

Proof. The theorem follows by combining Theorem 61 and Theorem 58. □

References

[AF09] Luis Filipe Coelho Antunes and Lance Fortnow. "Worst-Case Running Times for Average-Case Algorithms". In: *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*. IEEE Computer Society, 2009, pp. 298–303. DOI: 10.1109/CCC.2009.12.

[Ben+92] Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. "On the Theory of Average Case Complexity". In: *J. Comput. Syst. Sci.* 44.2 (1992), pp. 193–219. DOI: 10.1016/0022-0000(92)90019-F.

[BFP05] Harry Buhrman, Lance Fortnow, and Aduri Pavan. "Some Results on Derandomization". In: *Theory Comput. Syst.* 38.2 (2005), pp. 211–227. DOI: 10.1007/s00224-004-1194-y.

[BT06a] Andrej Bogdanov and Luca Trevisan. "Average-Case Complexity". In: *Found. Trends Theor. Comput. Sci.* 2.1 (2006). DOI: 10.1561/0400000004.

[BT06b] Andrej Bogdanov and Luca Trevisan. "On Worst-Case to Average-Case Reductions for NP Problems". In: *SIAM J. Comput.* 36.4 (2006), pp. 1119–1159. DOI: 10.1137/S0097539705446974.

[CHV22] Lijie Chen, Shuichi Hirahara, and Neekon Vafa. "Average-Case Hardness of NP and PH from Worst-Case Fine-Grained Assumptions". In: *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*. Ed. by Mark Braverman. Vol. 215. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 45:1–45:16. DOI: 10.4230/LIPIcs.ITCS.2022.45.

[GK22] Halley Goldberg and Valentine Kabanets. "A Simpler Proof of the Worst-Case to Average-Case Reduction for Polynomial Hierarchy via Symmetry of Information". In: *Electron. Colloquium Comput. Complex.* 7 (2022), pp. 1–14.

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for All One-Way Functions". In: *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*. STOC '89. Seattle, Washington, USA: Association for Computing Machinery, 1989, pp. 25–32. ISBN: 0897913078. DOI: 10.1145/73007.73010.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001. ISBN: 0-521-79172-3. DOI: 10.1017/CBO9780511546891.

[Hir20] Shuichi Hirahara. "Characterizing Average-Case Complexity of PH by Worst-Case Meta-Complexity". In: *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*. Ed. by Sandy Irani. IEEE, 2020, pp. 50–60. DOI: 10.1109/FOCS46700.2020.00014.

[Hir21a] Shuichi Hirahara. "Average-case hardness of NP from exponential worst-case hardness assumptions". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 292–302. DOI: 10.1145/3406325.3451065.

[Hir21b] Shuichi Hirahara. "Symmetry of Information in Heuristica". Manuscript. 2021.

[HN21] Shuichi Hirahara and Mikito Nanashima. "On Worst-Case Learning in Relativized Heuristica". In: *Symposium on Foundations of Computer Science* (FOCS). 2021.

[HS22a] Shuichi Hirahara and Rahul Santhanam. "Errorless Versus Error-Prone Average-Case Complexity". In: *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*. Ed. by Mark Braverman. Vol. 215. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 84:1–84:23. DOI: 10.4230/LIPIcs.ITCS.2022.84.

[HS22b] Shuichi Hirahara and Rahul Santhanam. "Excluding PH Pessiland". In: *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*. Ed. by Mark Braverman. Vol. 215. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 85:1–85:25. DOI: 10.4230/LIPIcs.ITCS.2022.85.

[IL90] Russell Impagliazzo and Leonid A. Levin. "No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random". In: *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*. IEEE Computer Society, 1990, pp. 812–821. DOI: 10.1109/FSCS.1990.89604.

[Imp11] Russell Impagliazzo. "Relativized Separations of Worst-Case and Average-Case Complexities for NP". In: *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, USA, June 8-10, 2011*. 2011, pp. 104–114. DOI: 10.1109/CCC.2011.34.

[Imp95] Russell Impagliazzo. "A Personal View of Average-Case Complexity". In: *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*. 1995, pp. 134–147. DOI: 10.1109/SCT.1995.514853.

[IW97] Russell Impagliazzo and Avi Wigderson. "$P = BPP$ if E Requires Exponential Circuits: Derandomizing the XOR Lemma". In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*. Ed. by Frank Thomson Leighton and Peter W. Shor. ACM, 1997, pp. 220–229. DOI: 10.1145/258533.258590.

[KL18] Pravesh K. Kothari and Roi Livni. "Improper Learning by Refuting". In: *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*. Ed. by Anna R. Karlin. Vol. 94. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 55:1–55:10. DOI: 10.4230/LIPIcs.ITCS.2018.55.

[KSS94] Michael J. Kearns, Robert E. Schapire, and Linda Sellie. "Toward Efficient Agnostic Learning". In: *Mach. Learn.* 17.2-3 (1994), pp. 115–141. DOI: 10.1007/BF00993468.

[Lau83] Clemens Lautemann. "BPP and the Polynomial Hierarchy". In: *Inf. Process. Lett.* 17.4 (1983), pp. 215–217. DOI: 10.1016/0020-0190(83)90044-3.

[Lee06] Troy Lee. "Kolmogorov complexity and formula lower bounds". PhD thesis. University of Amsterdam, 2006.

[Lev74] Leonid A. Levin. "Laws of information conservation (nongrowth) and aspects of the foundation of probability theory". In: *Problemy Peredachi Informatsii* 10.3 (1974), pp. 30–35.

[Lev86] Leonid A. Levin. "Average Case Complete Problems". In: *SIAM J. Comput.* 15.1 (1986), pp. 285–286. DOI: 10.1137/0215020.

[LO21] Zhenjian Lu and Igor C. Oliveira. "An Efficient Coding Theorem via Probabilistic Representations and Its Applications". In: *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*. Ed. by Nikhil Bansal, Emanuela Merelli, and James Worrell. Vol. 198. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 94:1–94:20. DOI: 10.4230/LIPIcs.ICALP.2021.94.

[LOS21] Zhenjian Lu, Igor C. Oliveira, and Rahul Santhanam. "Pseudodeterministic algorithms and the structure of probabilistic time". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 303–316. DOI: 10.1145/3406325.3451085.

[LOZ22] Zhenjian Lu, Igor C. Oliveira, and Marius Zimand. "Optimal coding theorems in time-bounded Kolmogorov complexity". In: *49th International Colloquium on Automata, Languages and Programming*. 2022.

[LP20] Yanyi Liu and Rafael Pass. "On One-way Functions and Kolmogorov Complexity". In: *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*. 2020, pp. 1243–1254. DOI: 10.1109/FOCS46700.2020.00118.

[LP21a] Yanyi Liu and Rafael Pass. "On One-way Functions from NP-Complete Problems". In: *Electron. Colloquium Comput. Complex.* (2021), p. 59.

[LP21b] Yanyi Liu and Rafael Pass. "On the Possibility of Basing Cryptography on EXP$\neq$ BPP". In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part I*. 2021, pp. 11–40. DOI: 10.1007/978-3-030-84242-0_2.

[LV19a] Ming Li and Paul M. B. Vitányi. *An introduction to Kolmogorov complexity and its applications*. Springer, 2019.

[LV19b] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. ISBN: 978-3-030-11297-4. DOI: 10.1007/978-3-030-11298-1.

[LW95] Luc Longpré and Osamu Watanabe. "On Symmetry of Information and Polynomial Time Invertibility". In: *Inf. Comput.* 121.1 (1995), pp. 14–22. DOI: 10.1006/inco.1995.1120.

[MVW99] Peter Bro Miltersen, N. V. Vinodchandran, and Osamu Watanabe. "Super-Polynomial Versus Half-Exponential Circuit Size in the Exponential Hierarchy". In: *Computing and Combinatorics, 5th Annual International Conference, COCOON '99, Tokyo, Japan, July 26-28, 1999, Proceedings*. Ed. by Takao Asano, Hiroshi Imai, D. T. Lee, Shin-Ichi Nakano, and Takeshi Tokuyama. Vol. 1627. Lecture Notes in Computer Science. Springer, 1999, pp. 210–220. DOI: 10.1007/3-540-48686-0_21.

[NRS95] Ashish V. Naik, Kenneth W. Regan, and D. Sivakumar. "On Quasilinear-Time Complexity Theory". In: *Theor. Comput. Sci.* 148.2 (1995), pp. 325–349. DOI: 10.1016/0304-3975(95)00031-Q.

[NW94] Noam Nisan and Avi Wigderson. "Hardness vs Randomness". In: *J. Comput. Syst. Sci.* 49.2 (1994), pp. 149–167. DOI: 10.1016/S0022-0000(05)80043-1.

[Oli19] Igor C. Oliveira. "Randomness and Intractability in Kolmogorov Complexity". In: *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*. Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 32:1–32:14. DOI: 10.4230/LIPIcs.ICALP.2019.32.

[OS17] Igor C. Oliveira and Rahul Santhanam. "Pseudodeterministic constructions in subexponential time". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*. Ed. by Hamed Hatami, Pierre McKenzie, and Valerie King. ACM, 2017, pp. 665–677. DOI: 10.1145/3055399.3055500.

[RS21] Hanlin Ren and Rahul Santhanam. "Hardness of KT Characterizes Parallel Cryptography". In: *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*. 2021, 35:1–35:58. DOI: 10.4230/LIPIcs.CCC.2021.35.

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014. ISBN: 978-1-10-705713-5.

[SU05] Ronen Shaltiel and Christopher Umans. "Simple extractors for all min-entropies and a new pseudorandom generator". In: *J. ACM* 52.2 (2005), pp. 172–216. DOI: 10.1145/1059513.1059516.

[TV07] Luca Trevisan and Salil P. Vadhan. "Pseudorandomness and Average-Case Complexity Via Uniform Reductions". In: *Computational Complexity* 16.4 (2007), pp. 331–364.

[Vad17] Salil P. Vadhan. "On Learning vs. Refutation". In: *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*. Ed. by Satyen Kale and Ohad Shamir. Vol. 65. Proceedings of Machine Learning Research. PMLR, 2017, pp. 1835–1848.

[Vio05] Emanuele Viola. “On Constructing Parallel Pseudorandom Generators from One-Way Functions”. In: *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*. 2005, pp. 183–197. DOI: 10.1109/CCC.2005.16.

A Additional Properties of pK^t Complexity

A.1 Upper bound on the complexity of estimating pK^t

The following says that the problem of estimating the pK^t complexity of a given string for a given t is in promise AM.

Definition 64 (Gap-MINpKT)**.** For a function $\tau \colon \mathbb{N} \to \mathbb{N}$, let $\mathsf{Gap}_\tau\text{-}\mathsf{MINpKT}$ be the following promise problem $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$:

$$\Pi_{\mathsf{YES}} := \left\{ \left(x, 1^s, 1^t\right) \mid \mathsf{pK}^t(x) \le s \right\},$$
$$\Pi_{\mathsf{NO}} := \left\{ \left(x, 1^s, 1^t\right) \mid \mathsf{pK}^{\tau(t)}(x) > s + \log \tau(t) \right\}.$$

Proposition 65. *There is a constant $c \ge 1$ for which the following holds. For any function τ such that $\tau(t) \ge c \cdot t$, $\mathsf{Gap}_\tau\text{-}\mathsf{MINpKT} \in \mathsf{promiseAM}$.*

Proof. We first observe that the following problem $(\Pi'_{\mathsf{YES}}, \Pi'_{\mathsf{NO}})$ is in $\mathsf{promiseAM}$.

$$\Pi'_{\mathsf{YES}} = \left\{ \left(x, 1^s, 1^t\right) \mid \mathsf{pK}^t_{2/3}(x) \le s \right\},$$
$$\Pi'_{\mathsf{NO}} = \left\{ \left(x, 1^s, 1^t\right) \mid \mathsf{pK}^t_{1/3}(x) > s \right\}.$$

To solve the above problem, on input x, the verifier first sends a random $w \in \{0,1\}^t$ to the prover, who then sends back a program $\mathcal{M}_w$. Finally, the verifier accepts if and only if $|\mathcal{M}_w| \le s$ and $\mathcal{M}_w(w)$ outputs x within t steps. It is easy to see that if $x \in \Pi'_{\mathsf{YES}}$, then the above protocol accepts with probability at least $2/3$. If $x \in \Pi'_{\mathsf{NO}}$, then the fraction of $w \in \{0,1\}^t$ such that x can be generated by some size-s program in time t is less than $1/3$ (otherwise $\mathsf{pK}^t_{1/3}(x)$ would be at most s), so the protocol accepts with probability less than $1/3$. Now the proposition follows from Lemma 21, which implies that the set Π_{NO} of NO instances in $\mathsf{Gap}_\tau\text{-}\mathsf{MINpKT}$ satisfies $\Pi_{\mathsf{NO}} \subseteq \Pi'_{\mathsf{NO}}$, provided that c is a large enough constant. □

A.2 Relations between time-bounded Kolmogorov complexity notions

In this section, we present the tight relations between K^t, rK^t, and pK^t (see Section 2.5) that hold under derandomization assumptions.

Proposition 66. *The following results hold.*

- *If $\mathsf{E} \not\subseteq \mathsf{i.o.SIZE}\left[2^{\Omega(n)}\right]$, then there is a polynomial p such that $\mathsf{K}^{p(t)}(x) \le \mathsf{rK}^t(x) + \log p(t)$, for every t and x.*
- *If $\mathsf{E} \not\subseteq \mathsf{i.o.NSIZE}\left[2^{\Omega(n)}\right]$, then there is a polynomial p such that $\mathsf{K}^{p(t)}(x) \le \mathsf{pK}^t(x) + \log p(t)$, for every t and x.*

- *If* $\mathsf{BPE} \not\subseteq \mathsf{i.o.NSIZE}\left[2^{\Omega(n)}\right]$*, then there is a polynomial p such that* $\mathsf{rK}^{p(t)}(x) \le \mathsf{pK}^t(x) + \log p(t)$*, for every t and x.*

Proof. Let x be any string $\{0,1\}^n$ and t be any integer such that $t \ge n$.

For the first item, note that the assumption $\mathsf{E} \not\subseteq \mathsf{i.o.SIZE}\left[2^{\Omega(n)}\right]$ implies that there is a PRG $G\colon \{0,1\}^{O(\log s)} \to \{0,1\}^s$ that $(1/s)$-fools size-s circuits and has running time $\mathsf{poly}(s)$ [IW97]. Suppose $\mathsf{rK}^t(x) \le k$. Let $\mathcal{M} \in \{0,1\}^k$ be a randomized program with running time t that outputs x with probability at least $2/3$. Consider the following function C on inputs of length t:

$$C(w) = 1 \iff \mathcal{M}(w) = x.$$

It is clear that C can be implemented as a $\mathsf{poly}(t)$-size circuit, and by definition the acceptance probability of C is at least $2/3$. Then there exists some seed $z \in \{0,1\}^{O(\log t)}$ such that $C(G(z)) = 1$, which implies $\mathcal{M}(G(z)) = x$. This means given $\mathcal{M}$ and z, we can deterministically compute x in time $\mathsf{poly}(t)$. In other words, $\mathsf{K}^{\mathsf{poly}(t)} \le k + O(\log t)$.

Next, we show the second item. The assumption $\mathsf{E} \not\subseteq \mathsf{i.o.NSIZE}\left[2^{\Omega(n)}\right]$ implies that there is a PRG $G'\colon \{0,1\}^{O(\log s)} \to \{0,1\}^s$ that $(1/s)$-fools size-s nondeterministic circuits and has running time $\mathsf{poly}(s)$ [SU05]. Now suppose $\mathsf{pK}^t(x) \le k$. Consider the following function C' on t bits.

$$C'(w) = 1 \iff \exists\, \mathcal{M} \in \{0,1\}^k \text{ such that } \mathcal{M}(w) \text{ outputs } x \text{ within } t \text{ steps.}$$

It is easy to see that C' can be implemented as a size-$\mathsf{poly}(t)$ nondeterministic circuit. Then there exists some seed $z \in \{0,1\}^{O(\log t)}$ such that $C'(G'(z)) = 1$. This means for such z, there exists some program $\mathcal{M}' \in \{0,1\}^k$ (which can depend on z) such that $\mathcal{M}'(G'(z))$ runs t steps and outputs x. Then given such z and $\mathcal{M}'$, we can computes x in time $\mathsf{poly}(t)$, by first computing $w := G'(z)$ and executing $\mathcal{M}(w)$. This yields $\mathsf{pK}^{\mathsf{poly}(t)}(x) \le k + O(\log t)$.

The proof of the third item is essentially the same as that of the second item. The only difference here is that the assumption $\mathsf{BPE} \not\subseteq \mathsf{i.o.NSIZE}\left[2^{\Omega(n)}\right]$ implies a *pseudodeterministic* PRG that fools size-s nondeterministic circuits with seed length $O(\log s)$, which follows from the construction of the PRG in [SU05]. Therefore, at the end of the proof for the second item above, we can successfully compute $w := G'(z)$ with high probability (over the internal randomness of the algorithm computing G'), and hence we get $\mathsf{rK}^{\mathsf{poly}(t)}(x) \le k + O(\log t)$. □

B Probabilistic Fine-Grained Reduction for $\mathsf{UTIME}\left[2^{O(\sqrt{n \log n})}\right]$

Theorem 67. $\mathsf{NQL} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL} \implies \mathsf{UTIME}\left[2^{O(\sqrt{n \log n})}\right] \subseteq \mathsf{RTIME}\left[2^{O(\sqrt{n \log n})}\right]$.

We will need the following fine-grained version of Lemma 27.

Lemma 68. *If* $\mathsf{NQL} \times \mathsf{QLSamp} \subseteq \mathsf{AvgBPQL}$*, then there exists a polynomial τ such that the following promise problem is in* $\mathsf{promiseBPP}$*:*

$$\Pi_{\mathsf{YES}} := \left\{ (x, 1^s, 1^t) \mid \mathsf{pK}^t(x) \le s \right\},$$
$$\Pi_{\mathsf{NO}} := \left\{ (x, 1^s, 1^t) \mid \mathsf{pK}^{\widetilde{O}(t)\cdot\tau(|x|)}(x) > s + \log \tau(t) \right\}.$$

Proof. Define

$$L := \left\{ (\mathsf{DP}_k(x;z), w, 1^s, 1^n) \,\middle|\, \exists \mathcal{M} \in \{0,1\}^s,\ \mathcal{M}(w) \text{ outputs } x \in \{0,1\}^n \text{ within } |w| \text{ steps} \right\},$$

where $c > 0$ is some large constant so that $L \in \mathsf{NQL}$. Define a distribution family

$$D := \left\{ D_{\langle nk+k,t,s,n\rangle} \right\},$$

each member of which does the following: sample $u \sim \mathcal{U}_{nk+k}$ and $w \sim \mathcal{U}_t$ and then output $(u, w, 1^s, 1^n)$. (Again, for the simplicity of presentation, we omit the padding, which is of length at most $\widetilde{O}(t) \cdot \mathsf{poly}(nk)$.) By assumption, $(L, D) \in \mathsf{AvgBPQL}$. Let B be a randomized heuristic algorithm for (L, D) as described in Lemma 6. Now, define an algorithm B':

> On input $(x, 1^s, 1^t)$ with $x \in \{0,1\}^n$, set $k = s + 10$, sample $z \sim \mathcal{U}_{nk}$ and $w \sim \mathcal{U}_t$, and then output $B(\mathsf{DP}_k(x;z), w, 1^s, 1^n)$.

Note that B' runs in time

$$t_D := \widetilde{O}(t) \cdot p(n)$$

for some polynomial p. Below, we show that B' solves the mentioned promise problem correctly with high probability in the worst case.

First, consider the case that $(x, 1^s, 1^t) \in \Pi_{\mathsf{YES}}$. By the definitions of L and pK, for any choice of $z \in \{0,1\}^{nk}$,

$$\Pr_{w}\left[(\mathsf{DP}_k(x;z), w, 1^s, 1^n) \in L\right] \geq 2/3.$$

The definition of B then implies that

$$\Pr_{w,z,r_B}\left[B(\mathsf{DP}_k(x;z), w, 1^s, 1^n) = 1\right] > 1/2,$$

and so

$$\Pr_{r_{B'}}\left[B'(x, 1^s, 1^t) = 1\right] > 1/2, \tag{42}$$

where r_B denotes the internal randomness of B, and $r_{B'} = (w, z, r_B)$ that of B'.

Now consider the case that $(x, 1^s, 1^t) \in \Pi_{\mathsf{NO}}$. For a contradiction, suppose that

$$\Pr_{w,z,r_B}\left[B(\mathsf{DP}_k(x;z), w, 1^s, 1^n) = 1\right] > 1/3. \tag{43}$$

Recall $k = s + 10$. By a counting argument, for randomly selected u and w,

$$\Pr_{u,w}\left[(u, w, 1^s, 1^n) \in L\right] \leq \frac{2^s \cdot 2^{nk} \cdot 2^t}{2^{nk+k+t}} = \frac{1}{10}.$$

Then by definition of B,

$$\Pr_{u,w,r_B}\left[B(u, w, 1^s, 1^n) = 1\right] \leq 1/9. \tag{44}$$

Comparing Equations (43) and (44), we see that $B(-,\mathcal{U}_t,1^s,1^n)$ (2/9)-distinguishes the distribution $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ from uniform. Lemma 22 implies that

$$\begin{aligned}\mathsf{pK}^{\widetilde{O}(t)\cdot\tau_0(n)}(x) &\leq k+\log\tau_0(t)\\ &= s+10+\log\tau_0(t),\end{aligned}$$

for some polynomial τ_0 that depends on the polynomials p and p_{DP} from Lemma 22. This means that $(x,1^s,1^t)$ is *not* in Π_{NO}, by setting τ properly. This gives the desired contradiction. By definition of B', we have that

$$\Pr_{r_{B'}}\left[B'(x,1^s,1^t)=1\right]\leq 1/3. \tag{45}$$

By Equations (13) and (16), B' yields a $\mathsf{promiseBPP}$ algorithm for $(\Pi_{\mathsf{YES}},\Pi_{\mathsf{NO}})$ via standard error reduction techniques. □

The following is a simple corollary of Lemma 47 and Lemma 20.

Lemma 69 (Fine-Grained Weak Symmetry of Information for pK^t)**.** *If* $\mathsf{NQL}\times\mathsf{QLSamp}\subseteq\mathsf{AvgBPQL}$*, then there exist polynomials* p_0 *and* p *such that for every sufficiently large* $x\in\{0,1\}^n$*,* $m\in\mathbb{N}$ *and* $p_0(n,m)\leq t\leq 2^{n+m}$*,*

$$\Pr_{u\sim\{0,1\}^m}\left[\mathsf{pK}^t(x\circ u)>\mathsf{pK}^{t\cdot p(nm)}(x)+m-\log p(t)\right]\geq 0.99.$$

We will use the following black-box hitting set generator whose outputs can be small relative to the "hard string" used in its construction.

Lemma 70 ([Hir20, Theorem 4.3])**.** *For any* $T,m\in\mathbb{N}$ *with* $m\leq 2T$*, there exists a function* $H_{T,m}\colon\{0,1\}^T\times\{0,1\}^d\to\{0,1\}^m$ *and a deterministic procedure* $\mathsf{Recon}^{(-)}\colon\{0,1\}^a\to\{0,1\}^T$*, where* $d=O\big(\log T+\log^3 m\big)$ *and* $a=2m+O\big(\log T+\log^3 m\big)$ *such that, for any* $x\in\{0,1\}^T$ *and any function* $D\colon\{0,1\}^m\to\{0,1\}$ *that* 0.1*-avoids* $H_{T,m}(x,-)$*, there exists advice* $\alpha\in\{0,1\}^a$ *such that* $\mathsf{Recon}^D(\alpha)=x$*. Moreover,* $H_{T,m}$ *can be computed in time* $\mathsf{poly}(T)$ *and* Recon^D *can be computed in time* $\mathsf{poly}(T)$ *with oracle access to* D*.*

We are now ready to show Theorem 67.

Proof of Theorem 67. Let $L\in\mathsf{UTIME}[T(n)]$ with verifier V, where $T(n)=2^{O(\sqrt{n\log n})}$. Fix an input $x\in\{0,1\}^n$. Let $y_x\in\{0,1\}^{T(n)}$ be the unique L-witness for x. Let B be a $\mathsf{promiseBPP}$ algorithm that solves the promise problem $(\Pi_{\mathsf{YES}},\Pi_{\mathsf{NO}})$ from Lemma 68.

Let t be *any* number such that $q_0(T(n))\leq t\leq 2^{n/2}$, where q_0 is some polynomial specified later. Let $H:=H_{T(n),m}$ be the hitting set generator from Lemma 70, where $m:=m_x(t)\leq O(n)$ is defined later, and let $d:=O\big(\log T(n)+\log^3 m\big)$ be the seed length of H.

Using H and B, we will argue that the time-bounded pK complexity of y_x is small given x. First note the following.

Claim 71. *There exists a polynomial* q *such that for every* $q_0(T(n))\leq t\leq 2^{n/2}$ *and every* $z_H\in\{0,1\}^d$*,*

$$\mathsf{pK}^{t\cdot q(nm)}\left(x\circ H(y_x;z_H)\right)\leq\mathsf{pK}^t(x)+d+\log q(t).$$

Proof of Claim 71. Let L be the following language

$$L' := \left\{ \left(\mathsf{DP}_k(x \circ H(y; z'); z), w01^{T(n)^c}, 1^s, 1^n, 1^m\right) \;\middle|\; \exists \mathcal{M} \in \{0,1\}^s, \mathcal{M}(w) \text{ outputs } x \in \{0,1\}^n \text{ within } |w| \text{ steps, and } V(x,y) = 1 \right\},$$

where $c > 0$ is some constant so that $L' \in \mathsf{NQL}$. Define a distribution family

$$D := \left\{ D_{\langle (n+m)k+k, t+T(n)^c+1, s, n, m\rangle} \right\},$$

each member of which samples $u \sim \mathcal{U}_{(n+m)k+k}$, $w \sim \mathcal{U}_t$ and outputs

$$\left(u, w01^{T(n)^c}, 1^s, 1^n, 1^m\right).$$

By assumption, $(L', D) \in \mathsf{AvgBPQL}$. Let B' be a randomized heuristic algorithm for (L', D) as described in Lemma 6.

Define $s := \mathsf{pK}^t(x)$. By definition of pK^t and L', for any fixed $z \in \{0,1\}^{(n+m)k}$,

$$\Pr_w \left[\left(\mathsf{DP}_k(x \circ H(y_x; z_H); z), w01^{T(n)^c}, 1^s, 1^n, 1^m\right) \in L' \right] \geq 2/3.$$

Then by definition of B', it is easy to see that

$$\Pr_{z,w,r'_B} \left[B'\left(\mathsf{DP}_k(x \circ H(y_x; z_H); z), w01^{T(n)^c}, 1^s, 1^n, 1^m\right) = 1 \right] \geq 1/2. \tag{46}$$

On the other hand, by a counting argument it is easy to see that for u and w selected uniformly at random, we have

$$\begin{aligned} \Pr_{u,w} \left[\left(u, w01^{T(n)^c}, 1^s, 1^n, 1^m\right) \in L' \right] &\leq \frac{2^s \cdot 2^t \cdot 2^d \cdot 2^{(n+m)k}}{2^{|u|} \cdot 2^{|w|}} \\ &\leq 1/n. \end{aligned}$$

where the last inequality holds by setting $k := s + d + \log n$. By the definition of B', we have

$$\Pr_{u,w,r'_B} \left[B'\left(u, w01^{T(n)^c}, 1^s, 1^n, 1^m\right) = 1 \right] \leq o(1). \tag{47}$$

Given Equations (46) and (47), it is clear that $D(-) := B'\left(-, \mathcal{U}_t 01^{T(n)^c}, 1^s, 1^n, 1^m\right)$ is a randomized distinguisher for $\mathsf{DP}_k\left(x \circ H(y_x; z_H); \mathcal{U}_{(n+m)k}\right)$. Note that provided that q_0 is a large enough polynomial, D needs only $O(\log t)$ bits as advice, and then it runs in time

$$t_D := \widetilde{O}(t + T(n)^c) \cdot \mathsf{poly}(nm) \leq \widetilde{O}(t) \cdot \mathsf{poly}(nm).$$

Lemma 22 implies that

$$\mathsf{pK}^{\widetilde{O}(t) \cdot q(nm)}(x \circ H(y_x; z_H)) \leq k + O(\log t) \leq \mathsf{pK}^t(x) + d + \log q(t),$$

for some large polynomial q. This completes the proof of Claim 71. $\square$

By Lemma 69, there exist polynomial p_0 and p such that for every $p_0(n,m) \le t_1 \le 2^{n+m}$

$$\Pr_{u \sim \{0,1\}^m}\left[\mathsf{pK}^{t_1}(x \circ u) > \mathsf{pK}^{t_1 \cdot p(nm)}(x) + m - \log p(t_1)\right] \ge 0.99. \tag{48}$$

We set $t_1 := \widetilde{O}(t \cdot q(nm)) \cdot \tau(n+m)$, where τ is the function from Lemma 68. Note that we can make the polynomial q_0 to be large enough so that t_1 satisfies the condition for which Equation (48) holds. Now we choose m so that

$$\mathsf{pK}^{t_1 \cdot p(nm)}(x) + m - \log p(t_1) \ge s + \log \tau(t \cdot q(nm)). \tag{49}$$

That is, we set

$$m := \left(\mathsf{pK}^{t}(x) - \mathsf{pK}^{t \cdot n^c}(x)\right) + c \cdot \log t,$$

where $c > 0$ is some large constant, and we use the fact that $t \le 2^{n/2}$ to simplify t_1. Define the following (probabilistic) function:

$$D(-) := \neg B\left(x \circ -, 1^s, 1^{t \cdot q(nm)}\right).$$

Observe the following properties of D.

- For every $z \in \{0,1\}^d$, $\mathbf{Pr}_D\left[D(H(y_x; z)) = 0\right] \ge 2/3$. This follows from Claim 71 and the correctness of B on solving the promise problem $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ from Lemma 68.
- For at least 0.99 fraction of $\beta \in \{0,1\}^m$, $\mathbf{Pr}_D\left[D(\beta) = 1\right] \ge 2/3$. This follows from Equations (48) and (49) (and the correctness of B).
- D runs in time $\mathsf{poly}(t)$.

At this point, we can see that D is a randomized function that *distinguishes* $H(y_x; \mathcal{U}_d)$ from $\mathcal{U}_m$, and by randomly fixing its internal randomness we get a deterministic distinguisher with good probability. However, in order to utilize the reconstruction procedure in Lemma 70 to recover y_x, we need a deterministic function that *avoids* H. By amplifying the success probability of D on the "promised" inputs, we can show that randomly fixing the internal randomness of D gives a function that avoids H with high probability. More specifically, using standard error reduction techniques, we can make both the probability in the first two items above at least $1 - 1/(4 \cdot 2^m)$, which only causes a multiplicative overhead of $\mathsf{poly}(m)$ in the running time. By abusing notation, we use the same D to denote this new function with small error. Now consider the set S of inputs for which D has this small error. That is,

$$S := \left\{H(y_x; z) \mid z \in \{0,1\}^d\right\} \cup \left\{\beta \in \{0,1\}^m \mid \Pr_D[D(\beta) = 1] \ge 1 - 1/(4 \cdot 2^m)\right\}.$$

Note that $|S| \le 2^m$. For $\beta \in S$, let us say that the *correct* answer of β is 0 (resp. 1) if β is in the first (resp. second) subset in the definition of S. By a union bound, we have

$$\Pr_{r_D \sim \{0,1\}^{\mathsf{poly}(t)}}\left[\exists\, \beta \in S \text{ such that } D(\beta; r_D) \text{ is not correct}\right] \le |S| \cdot \frac{1}{4 \cdot 2^m} \le \frac{1}{4}, \tag{50}$$

where r_D above denotes the internal randomness of D. This means with probability at least 2/3 over a uniformly random r_D, $D(-; r_D)$ is a deterministic function that is correct on *all* the inputs in S. In particular, such a function 0.1-avoids H. Let us say that r_D is *good* if this is true. Let $\mathsf{Recon}^{(-)}\colon \{0,1\}^a \to \{0,1\}^{T(n)}$ be the reconstruction procedure in Lemma 70, where $a := 2m + O\big(\log T + \log^3 m\big)$. We have

$$\begin{aligned}
&\mathop{\mathbf{Pr}}_{r_D\sim\{0,1\}^{\mathsf{poly}(t)}}\left[\exists\, \alpha \in \{0,1\}^a \text{ such that } \mathsf{Recon}^{D(-;r_D)}(\alpha) = x\right] \\
&\geq \mathop{\mathbf{Pr}}_{r_D}\left[\exists\, \alpha \in \{0,1\}^a \text{ such that } \mathsf{Recon}^{D(-,r_D)}(\alpha) = x \,\middle|\, r_D \text{ is good}\right] \cdot \mathop{\mathbf{Pr}}_{r_D}\left[r_D \text{ is good}\right] \\
&\geq \frac{2}{3}. && \text{(Lemma 70 and Equation (50))}
\end{aligned}$$

Note that given x, D can be constructed using the integers s, t, n, m and the code for B, which can be encoded using $O(\log t)$ bits. Then the above implies

$$\mathsf{pK}^{\mathsf{poly}(t\cdot T)}\,(y_x \mid x) \leq a + O(\log t) \leq 2m + O\big(\log t + \log^3 m\big)\,. \tag{51}$$

Note that the above holds for every $q_0(T(n)) \leq t \leq 2^{n/2}$. Also, recall that $m = \mathsf{pK}^t(x) - \mathsf{pK}^{t\cdot c}(x) + c \cdot \log t$ for some constant $c > 0$. By Lemma 46, there exists some t_* such that

$$q_0(T(n)) \leq t_* \leq q_0(T(n)) \cdot 2^{c\cdot\sqrt{n\log n}} \leq 2^{O(\sqrt{n\log n})} \quad \text{and} \quad \mathsf{pK}^{t_*}(x) - \mathsf{pK}^{t_*\cdot n^c}(x) \leq O\Big(\sqrt{n\log n}\Big)\,.$$

Then for such t_*, we have $m \leq O\big(\sqrt{n\log n}\big)$. Therefore, plugging this t_* into Equation (51) for t, we have that there exists some constant $d > 0$ such that

$$\mathsf{pK}^{2^{d\cdot\sqrt{n\log n}}}(y_x \mid x) \leq d \cdot \sqrt{n\log n}.$$

This suggests the following probabilistic algorithm A for solving L:

> On input $x \in \{0,1\}^n$, A samples $w \sim \{0,1\}^{d\cdot\sqrt{n\log n}}$. It then exhaustively searches over all $\mathcal{M} \in \{0,1\}^{d\cdot\sqrt{n\log n}}$, running $\mathcal{M}(w,x)$ for $2^{d\cdot\sqrt{n\log n}}$ steps to produce some output y. A accepts iff a y is obtained such that $V(x,y) = 1$.

This completes the proof. □

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il